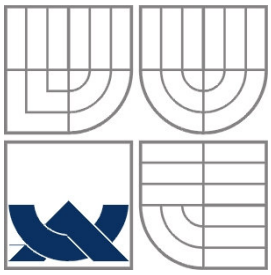
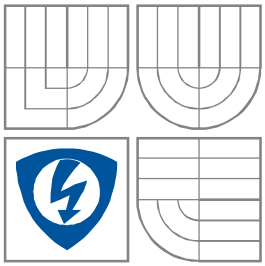


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

PŘÍDAVNÝ DISPLEJ LCD K LABORATORNÍMU PŘÍPRAVKU S PROGRAMOVATELNÝM OBVODEM

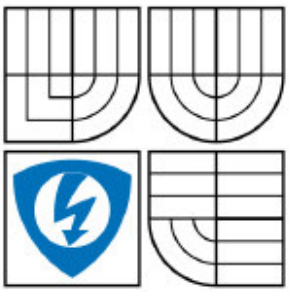
ADDITIONAL LCD Display for Laboratory Kit with a Programmable Device

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. Jaroslav Pajskr
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Jaromír Kolouch, CSc.
SUPERVISOR

BRNO, 2008



VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav radioelektroniky

Diplomová práce

magisterský navazující studijní obor
Elektronika a sdělovací technika

Student:	Pajskr Jaroslav Bc.	***ID:***	89227
Ročník:	2	***Akademický rok:***	2007/2008

NÁZEV TÉMATU:

Přídavný displej LCD k laboratornímu přípravku s programovatelným obvodem

POKYNY PRO VYPRACOVÁNÍ:

Zpracujte přehled displejů LCD vhodných k použití jako periferie k zobrazování údajů o stavu číslicových systémů. Prostudujte podrobně způsoby řízení vybraného typu displeje a vypracujte příslušné algoritmy.

Zpracujte návrh zapojení přídavné desky s displejem pro připojení k laboratornímu přípravku s programovatelným obvodem (PLD, FPGA) a podklady pro výrobu desky.

Oživte vyrobenou desku, vytvořte odpovídající blok pro řízení displeje v programovatelném obvodu a ověřte funkci displeje zpracováním příkladů jeho použití včetně jejich implementace.

DOPORUČENÁ LITERATURA:

[1] MATOUŠEK, D. Práce s inteligentními displeji LCD. Praha: BEN - technická literatura, 2006.

[2] KOLOUCH, J. Programovatelné logické obvody a návrh jejich aplikací v jazyku VHDL. Skriptum. Brno: FEKT VUT v Brně, 2006.

[3] KOLOUCH, J. Programovatelné logické obvody - přednášky. Skriptum. Brno: FEKT VUT v Brně, 2006.

Termín zadání:	5.10.2007	***Termín odevzdání:***	30.5.2008

Vedoucí práce: doc. Ing. Jaromír Kolouch, CSc.

prof. Dr. Ing. Zbyněk Raida
předseda oborové rady

UPOZORNĚNÍ:

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Bc. Jaroslav Pajskr
Bytem: Husova 1402, Hradec Králové 8, 500 08
Narozen/a (datum a místo): 24. prosince 1983 v Hradci Králové

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 53, Brno, 602 00
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Dr. Ing. Zbyněk Raida, předseda rady oboru Elektronika a sdělovací technika
(dále jen „nabyvatel“)

Čl. 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):

   ☐ disertační práce
   ☒ diplomová práce
   ☐ bakalářská práce
   ☐ jiná práce, jejíž druh je specifikován jako .....
   (dále jen VŠKP nebo dílo)

   Název VŠKP: Přídavný displej LCD k laboratornímu přípravku s programovatelným obvodem
   Vedoucí/ školitel VŠKP: doc. Ing. Jaromír Kolouch, CSc.
   Ústav: Ústav radioelektroniky
   Datum obhajoby VŠKP: _____

   VŠKP odevzdal autor nabyvateli*:

   ☒ v tištěné formě – počet exemplářů: 2
   ☒ v elektronické formě – počet exemplářů: 2

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.

3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.

4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

* hodící se zaškrtněte

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.

2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.

3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti

   - ☒ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy (z důvodu utajení v něm obsažených informací)

4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.

2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.

3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.

4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: 30. května 2008

…….. ……

Nabyvatel Autor

Abstrakt

Důležitou součástí číslicového zařízení je rozhraní, kterým uživatel může kontrolovat stav zařízení, případně ho nastavovat. Existuje mnoho způsobů realizace takového rozhraní. Většinou se volí takové řešení, které postačuje a je nejjednodušší. Nejčastěji je rozhraní realizováno pomocí displeje. Tato práce se zabývá návrhem desky pro připojení vhodného LCD displeje k programovatelnému obvodu, návrhem ovládacího algoritmu displeje a návrhem jednoduchého rozhraní pro ovládání displeje. V práci jsou dva návrhy ovládání displeje. První využívá procesor PicoBlaze, ve kterém jsou implementovány všechny požadované funkce. Druhý je realizován stavovými automaty napsanými jazyku VHDL, který je ekvivalentní k prvnímu návrhu, ale proti prvnímu návrhu je rychlejší a vyžaduje méně hardwarových prostředků.

Klíčová slova

Grafický LCD displej, Spartan 3, PicoBlaze, VHDL

Abstrakt

The main part of the digital application is its user interface. Users can check the status of the programme or change its state. There are many ways to obtain a suitable interface. During the design stage the simplest interface is chosen that provides the necessary functions. In most cases the interface contains a display. This diploma thesis deals with the design of an extension board to plug in a display to a programmable device, a control algorithm for the display and the design of a simple display interface.

There are two ways to design software. The first of them is achieved by the processor PicoBlaze, which contains all the required functions. The second solution is by the state machine written in VHDL language. Both solutions can be used in the same way, but the latter solution is quicker and requires less hardware resources.

Key words:

Graphical LCD display, Spartan 3, PicoBlaze, VHD

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Přídavný displej LCD k laboratornímu přípravku s programovatelným obvodem jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 30. května 2008 .....
podpis autora

Poděkování

Děkuji vedoucímu diplomové práce doc. Ing. Jaromíru Kolouchovi, CSc. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne 30. května 2008 .....
podpis autora

Bibliografická citace

PAJSKR, J. *Přídavný displej LCD k laboratornímu přípravku s programovatelným obvodem.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 49 s. Vedoucí diplomové práce doc. Ing. Jaromír Kolouch, CSc.

Obsah

1. Úvod

Důležitou součástí číslicového zařízení je rozhraní, kterým uživatel může kontrolovat stav zařízení, případně ho nastavovat. Existuje mnoho způsobů realizace takového rozhraní. Většinou se volí takové řešení, které postačuje a je nejjednodušší. Nejčastěji je rozhraní realizováno pomocí displeje. Tato práce se zabývá návrhem desky pro připojení vhodného LCD displeje k programovatelnému obvodu, návrhem ovládacího algoritmu displeje a návrhem jednoduchého rozhraní pro ovládání displeje.

2. Řadiče displejů LCD

Ovládání displeje tvořeného maticí bodů je hardwarově i časově náročné, proto se používají řadiče displejů. Tyto řadiče se ovládají pomocí sběrnice jednoduchými příkazy. Řadiče obsahují paměť, do které se ukládají hodnoty. Tyto hodnoty se pak pravidelně vypisují na displej. Z toho vyplývá, že řadič nepotřebuje pravidelně nastavovat, aby se na displeji zobrazily údaje, ale jen tehdy, je-li potřeba změnit zobrazené informace. To je jejich velká výhoda.

2.1. Druhy LCD displejů

Podle způsobu ovládání a následné zobrazení na displeji lze rozdělit řadiče do dvou základních kategorií.

Alfanumerické displeje

Nejrozšířenějším alfanumerickým řadičem je HD44780. Jeho ekvivalent je KS0066. S tímto řadičem se prodávají displeje od 8x2 znaků až do 40x4 znaků. Podrobnější informace o tomto řadiči jsou uvedeny v [7]. Obvod obsahuje paměť DDRAM o velikosti 80x8 bitů. V této paměti jsou uloženy adresy symbolů, které se zobrazí na displeji. Každé adrese v paměti DDRAM odpovídá pozice na displeji. Při běžné obsluze displeje se mění pouze data v této paměti. V paměti CGROM je uložena sada symbolů. Tato paměť je rozdělena na 2 části. Větší část o velikosti 9920 bitů je paměť ROM, ve které je uložena sada symbolů od výrobce. Podle označení řadiče lze poznat, jakou sadou je vybaven. Menší část paměti o velikosti 64 byte je paměť RAM. Slouží k tomu, aby mohl uživatel definovat 8 vlastních symbolů.

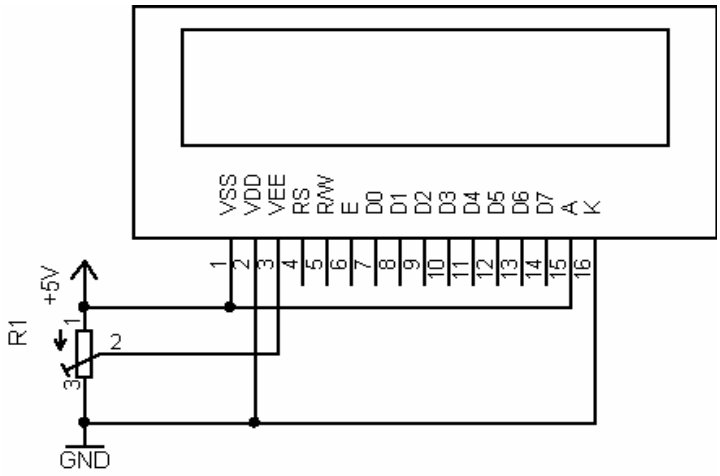
Grafické displeje

Grafických řadičů LCD displeje je více druhů. Liší se především velikostí zobrazovaného pole. Pro rozměr zobrazovaného pole 122x32 bodů je určen řadič SED1520. Obdobný obvod, pro rozměr 128x64, je řadič KS0107. Oba řadiče mají podobné ovládání. Obsahují stejně jako u alfanumerického řadiče paměť DDRAM, do které se ukládají zobrazovaná data. Tyto řadiče nemají paměť CGROM. Do paměti DDRAM se ukládají jednotlivé osmibitové sloupce, které vyjadřují zobrazený údaj. To znamená, že lze přímo vykreslit body na displej. Z toho ale vyplývá jedna nevýhoda. Pokud se bude na grafický displej psát text, musí se každé písmeno popsat 5 byty. Displeje s těmito řadiči mají rozdílně vyvedenou sběrnici a také mají rozdílné ovládací příkazy.

2.2. Zapojení LCD displejů

Alfanumerický displej CM1621 s řadičem HD44780

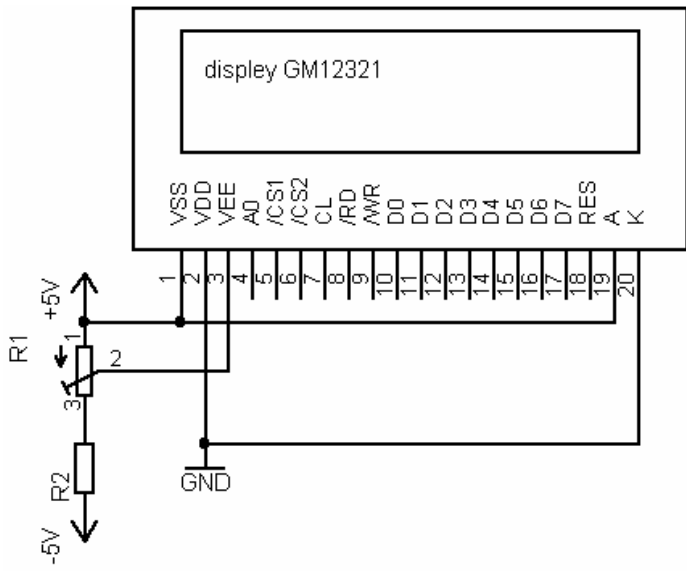
Displej s tímto řadičem má vyvedeno 16 vývodů. Lze ho napájet ze zdroje napětí od 4,75 V do 5,25 V. Základní zapojení displeje s napájecím zdrojem je na Obr. 1, kde k VSS a VDD se připojí napájecí napětí, VEE slouží k nastavení kontrastu, A a K je napájení podsvětlení displeje. Řídící signály jsou RS, R/W a E. Datová sběrnice je tvořena vývody D0 až D7. Podrobnější informace jsou uvedeny v [6] a [7].



Obr. 1: Základní zapojení alfanumerického displeje

Grafický displej GM12321 s řadičem SED1520

Tento typ displeje má 20 vývodů. Význam jednotlivých vývodů je na Obr. 2. Rozmezí napájení je od 4,5 V do 5,5 V. Napájení se přivádí na vývody VSS a VDD. Rezistorem R1 lze nastavovat kontrast. Obvod má vyveden reset, datovou sběrnici D0 až D7 a řídící sběrnici s vývody A0, CS1, CS2, CL, RD, WR. Vývodem A0 se rozlišují data od příkazu. Pomocí CS1 a CS2 se vybírá, se kterou částí displeje se bude pracovat. CL je externí taktování a RD a WR určují směr přenosu dat. Podrobnější informace o displeji a řadiči jsou uvedeny v [8] a [9].

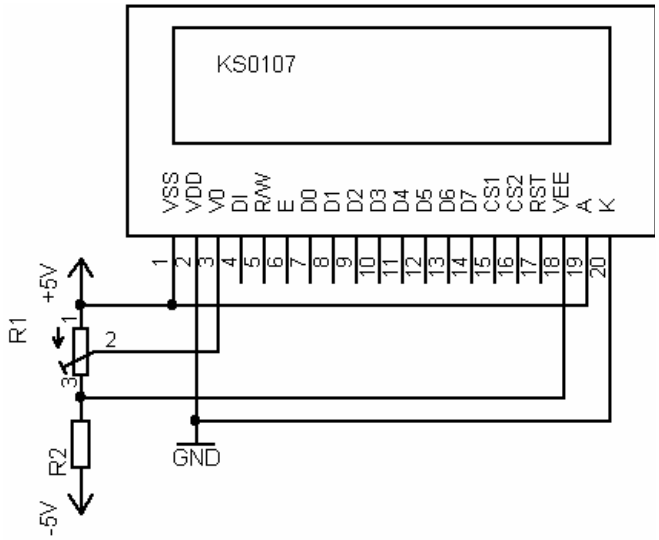


Obr. 2: Základní zapojení grafického displeje s řadičem SED1520

Grafický displej GM12641 s řadičem KS0107

Stejně jako displej s řadičem SED1520 má tento displej 20 vývodů. Na Obr. 3 je uveden příklad zapojení displeje. Displej potřebuje napájecí napětí v rozsahu od 4,5 V do 5,5 V. Napájení displeje je na vývodech VSS a VDD. Displej má vyvedeno záporné napětí, ze kterého se napájí LCD displej. Pomocí tohoto napětí se nastavuje kontrast na vývodu V0. Datovou sběrnici tvoří vývody D0 až D7. Řídící sběrnice má DI pro rozlišení příkazu od dat

a R/W pro rozlišení směru přenášených dat. Vývod E je taktování. Výběr části displeje zajistí vývody CS1 a CS2. Podrobnější informace o displeji a řadiči jsou uvedeny v [10] a [11].



Obr. 3: Základní zapojení grafického displeje s řadičem KS0107

2.3. Ovládání řadičů LCD displeje

Řadič HD44780

LCD displej s tímto řadičem má vyvedeno 16 vývodů. Datová sběrnice má 8 bitů. K řízení dat na datové sběrnici jsou 3 vývody. Pomocí vývodu E, změnou své hodnoty z 1 do 0, se převezmou data ze sběrnice a vykoná se instrukce. Vývod RS rozhoduje, jestli je přijímán příkaz nebo jsou přenášeny data. Řídící signál R/W určuje směr toku dat. Seznam instrukcí a další informace o displeji jsou uvedeny v [6] a [7]. Seznam instrukcí je v Tab. 1. Další informace k tomuto řadiči lze nalézt v [3].

Tab. 1: Seznam instrukcí řadiče HD44780

Instrukce	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Popis
Výmaz displeje	0	0	0	0	0	0	0	0	0	1	Vymaže displej a nastaví DDRAM na 0
Návrat na pozici 0	0	0	0	0	0	0	0	0	1	-	Nastaví DDRAM na 0, obsah displej se nezmění
Mód posunu	0	0	0	0	0	0	0	1	I/D	S	Nastavení směru pohybu kurzoru
Mód kurzoru	0	0	0	0	0	0	1	D	C	B	B – blikání kurzoru, C – zapnutí kurzoru, D - zapnutí displeje
Posun kurzoru	0	0	0	0	0	1	S/C	R/L	-	-	Pohyb kurzoru a posun dipleje bez změny obsahu DDRAM
Nastavení funkce	0	0	0	0	1	DL	N	F	-	-	bitů rozhraní–DL, počet řádek-N,rozměry pole -F
Nastavení CGRAM	0	0	0	1	CGRAM adresa						Data jsou poslána i přijata po tomto nastavení
Nastaví DDRAM	0	0	1	DDRAM adresa							Data jsou poslána i přijata po tomto nastavení

Čtení příznaku BF a adresy	0	1	BF	CGRAM/DDRAM adresa	BF – konec provádění příkazu, nastavená adresa
Zápis dat do paměti	1	0	Zapisovaná data		Zápíše data do CGRAM nebo DDRAM
čtení dat z paměti	1	1	Přečtená data		Přečte data z CGRAM nebo DDRAM

Význam zkratek:

I/D:	0 – zvyšování	1 – snižování
S:	0 – bez posunu displeje	1 – posun displeje při zápisu dat
D:	0 – vypnutí displeje	1 – zapnutí displeje
C:	0 – vypnutí kurzoru	1 – zapnutí kurzoru
B:	0 – vypnutí blikání displeje	1 – zapnutí blikání displeje
S/C:	0 – posun kurzoru	1 – posun displeje
R/L:	0 – posun doleva	1 – posun doprava
DL:	0 – 4 bitová komunikace	1 – 8 bitová komunikace
N:	0 – 1 řádek	1 – 2 řádky
F:	0 – 5x8 bodů	1 – 5x10 bodů
BF:	0 – operace je ukončena	1 – operace probíhá

Řadič SED1520

Tento řadič je uzpůsoben pro napojení k mikrokontroleru. Je vybaven řídícími signály /RD, /WR, /CS1 a /CS2, které umožňují displej připojit k externí datové a řídící sběrnici procesoru řady 80. K připojenému displeji se pak program chová jako k externí paměti. Změnou konfigurace lze LCD displej přizpůsobit procesoru řady 68.

Tento řadič obsahuje paměť DDRAM, ve které jsou uloženy data. Tato data vyjadřují body, které se zobrazí. Neprovádí se zde přiřazení pole bodů hodnotě paměti podle sady znaků, ale všechny znaky se musí přímo uložit do paměti DDRAM. To je určitá nevýhoda, protože obslužný program pro ovládání bude komplikovanější, ale díky tomuto systému lze definovat libovolné znaky. V řadiči lze definovat adresu prvního řádku – Display start address. Díky tomu lze jednoduše rolovat text na displeji. Protože tento řadič umí ovládat maximálně 72 sloupců, musí se celý displej o rozměrech 122x32 bodů rozdělit na 2 časti. Displej pak obsahuje 2 stejné řadiče. Aby se rozlišilo, se kterým řadičem se bude pracovat, jsou z displeje vyvedeny signály /CS1 a /CS2. Seznam instrukcí řadiče je uveden v Tab.2.

Tab. 2: Seznam instrukcí řadiče SED1520

Instrukce	A0	RD	WR	D7	D6	D5	D4	D3	D2	D1	D0	Popis
Vypnutí displeje	0	1	0	1	0	1	0	1	1	1	0/1	1: zapne displej 0: vypne displej
Počáteční řádek	0	1	0	1	1	0	Adresa počátku (0 -31)					Přiřadí adresu první zobrazované řádky v paměti
Adresa stránky	0	1	0	1	0	1	1	1	0	page (0-3)		Nastaví aktuální stránku
Nastavení sloupce	0	1	0	0	Adresa aktuálního sloupce (0 – 72)							Vloží do Column registru adresu aktuálního sloupce
Stavový registr	0	0	1	B	A	O	R	0	0	0	0	B: 0-čeká, 1-pracuje A: 0-invertované, 1-přímé O: 0-zap. diplej, 1-vyp. R: 0-dipl.mode, 1-reset

Zápis dat	1	1	0	Zapsaná data								Zapíše data do RAM
Čtení dat	1	0	1	Přečtená data								Přečte data z RAM
Volba ADC	0	1	0	1	0	1	0	0	0	0	0/1	0: CW –invert. zobr. 1: CCW- přímé zobrazení
Statické řízení	0	1	0	1	0	1	0	0	1	0	0/1	0: statické řízení 1: normální řízení
Pracovního cyklu	0	1	0	1	0	1	0	1	0	0	0/1	1: 1/32 0: 1/16
Čtení, zápis	0	1	0	1	1	1	0	0	0	0	0	Čtení/zápis je povoleno
Konec	0	1	0	1	1	1	0	1	1	1	0	Konec čtení/zápis
Reset	0	1	0	1	1	1	0	0	0	1	0	Softwarový reset

Datová sběrnice má pouze 8 bitů, proto se musí pole 72x32 bodů rozdělit na 4 stránky o rozměrech 72x8 bodů. Informace o tom, jaká stránka je vybraná je uložena v registru page. Adresa v registru Column address udává sloupec, na který se zapíší data.Tímto způsobem lze zobrazit kterýkoliv bod na displeji o rozměrech 122x32 bodů. V [8] a [9] jsou uvedeny podrobnější informace o displeji.

Řadič KS0107

Tento řadič je určený pro LCD displeje s polem 128x64 bodů. Je rozšířenou verzí řadiče SD1520. Displeje s těmito řadiči mají rozdílně zapojené vývody. Liší se i instrukce. U sadiče KS0107 není podpora napojení na sběrnici procesoru řady 80. Jeho řídící signály jsou R/W – volba směru posílaných dat, D/I – rozlišení, jestli se jedná o instrukci nebo o data, E – taktování, které zajišťuje čtení dat ze sběrnice po sestupné hraně. Obvod má méně ovládacích instrukcí. Řadič má podobné ovládání jako řadič SD1520. Také propojením dvou stejných řadičů se získá displej s dvojnásobným počtem bodů. Proto má displej s tímto řadičem vývody CS1 a CS2 pro volbu mezi jednou nebo druhou půlkou displeje. Aby řadič mohl ovládat 64 řádků, má celkem 8 stránek, do kterých se může zapsat 8x64 bitů dat. Seznam implementovaných instrukcí je uveden v Tab. 3.

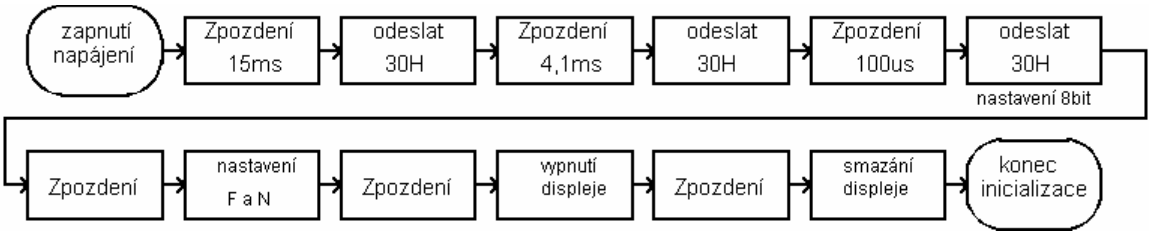
Tab. 3: Seznam instrukcí řadiče KS0107

Instrukce	D/I	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Popis
Vypnutí displeje	0	0	0	0	1	1	1	1	1	0/1	1: zapne displej 0: vypne displej
Počáteční řádek	0	0	0	1	Adresa počátku (0 -63)						Přiřadí adresu první zobrazované řádky v paměti
Adresa stránky	0	0	1	0	1	1	1	page (0-7)			Nastaví aktuální stránku
Nastavení sloupce	0	0	1	1	Adresa aktuálního sloupce (0 - 63)						Vloží do Column registru adresu aktuálního sloupce
Stavový registr	0	1	B	L	O	R	0	0	0	0	B: 0-čeká, 1-pracuje O: 0-zap. diplej, 1-vyp. R: 0-dipl.mode, 1-reset
Zápis dat	1	0	Zapsaná data								Zapíše data do RAM
Čtení dat	1	1	Přečtená data								Přečte data z RAM

2.4. Algoritmus inicializace a ovládání

Řadič HD44780

Před zahájením ovládání řadiče je vhodné po připojení napájecího napětí počkat 15ms. Poté se musí poslat displeji čtyři instrukce, které nastaví mód komunikace. Lze nastavit 4-bitovou nebo 8-bitovou komunikaci. Dále se při inicializaci posláním instrukce displej vypne. Následuje instrukce pro vymazání displeje. Vývojový diagram inicializace je na Obr. 4.

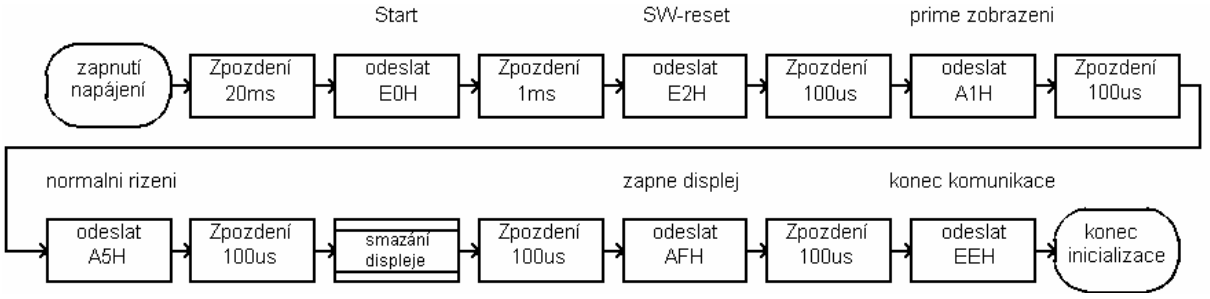


Obr. 4: Vývojový diagram inicializace displeje s řadičem HD44780

Po inicializaci se do paměti DDRAM uloží hodnoty, které budou zobrazeny. Podle konfigurace se pohybuje kurzor po displeji a nastavuje se adresa zápisu. Při vhodném nastavení lze nastavit adresu zápisu do místa, kde se má zobrazit první symbol a pak se opakovaným zápisem dat kurzor automaticky posouvá. Díky tomuto nastavení není potřeba měnit pozici zápisu při každé zapsané hodnotě.

Řadič SED1520

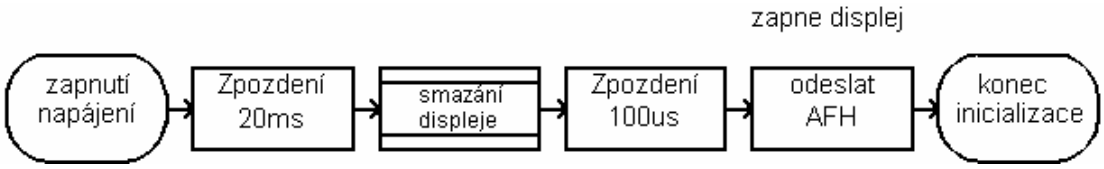
Tento řadič nemá uvedeno v katalogovém listu potřebnou dobu aktivace po připojení napájecího napětí. Proto je zvolena tato doba 20 ms. Tato hodnota se ověří při práci s displejem. Před zahájením jakékoliv komunikace s displejem se musí vyslat instrukce start a pro ukončení komunikace se musí vyslat instrukce konec. Po zotavení displeje se displej instrukcí zapne, další instrukcí se resetuje a poté se nastaví adresa první zobrazované řádky. Dále je vhodné nastavit způsob automatického nastavování adresy aktuálního sloupce. Nyní je možné zahájit načítání zobrazovaných dat do displeje. Zde se nejprve musí zvolit řádek, do kterého se bude zapisovat. Dále se musí nastavit adresa sloupce, do kterého se uloží data a poté se mohou začít posílat postupně data. Pokud se dojde na konec řádku a nezmění se adresa aktuálního řádku, dojde k postupnému přepisování už zapsaných dat. Proto se s tím musí počítat a na konci řádku nastavit adresu dalšího řádku. Po nastavení všech řádků a sloupců se ukončí komunikace instrukcí konec. Při nastavování zobrazení se nesmí zapomenout na to, že je displej rozdělen na 2 části, které se rozlišují vývody CS1 a CS2. To znamená, že pokud by se měnil celý jeden řádek, tak pro nastavení první poloviny musí být aktivní CS1 a pro dokončení řádku musí být aktivní CS2. Ovládání obou polovin displeje je stejné. Vývojový diagram inicializace řadiče je na Obr. 5.



Obr. 5: Vývojový diagram inicializace displeje s řadičem SED1520

Řadič KS0107

Hlavní rozdíl ovládání řadiče vůči předchozímu řadiči je v zahajování a ukončování komunikace. Tento řadič má výrazně méně instrukcí a zde už nejsou instrukce pro zahájení a ukončení komunikace. Zde se také může displej vypnout i nastavit adresu první zobrazované řádky, ale nelze tento řadič softwarově resetovat. Řadič má výrazně jednodušší ovládání. Po přivedení napájení se opět musí počkat, dokud se obvod nezotaví. Po zotavení se nastaví adresa aktuálního řádku a adresa aktuálního sloupce. Dále je také možné data uložit. V katalogovém listu [10] se nepíše o tom, že by se adresa aktuálního sloupce automaticky měnila po zápisu dat. Proto se zřejmě musí vždy zapsat adresa sloupce před každým zápisem dat. To je proti řadiči SED1520 drobná komplikace. Znamená to větší počet přenesených dat a tím i pomalejší změnu zobrazovaných dat. Vývojový diagram inicializace je na Obr. 6.



Obr. 6: Vývojový diagram inicializace displeje s řadičem KS0107

3. Návrh přípravku

3.1. Vývojový kit

SPARTAN 3

Základním obvodem vývojového kitu Spartan 3 je XC3S200. Tento obvod má několik částí. Mezi nejdůležitější patří IOB – I/O bloky, CLB – funkční bloky pro obecné použití, propojovací struktura a další speciální prvky. Mezi nejdůležitější speciální prvky patří bloky paměti RAM po 18 kb, celkem až 1872 kb. Dále bloky DLL pro kompenzaci zpoždění hodinových signálů. Obsahuje i hardwarové násobičky pro 18x18 bitů. Může mít maximálně 104 hardwarových násobiček.

Vývojový kit obsahuje několik dalších komponent. Má paměť PROM o velikosti 2 Mbitů. Další součástí je asynchronní paměť SRAM s velikostí 1 Mbyte. Přípravek je vybaven VGA portem s možností nastavení 8 barev. Umožňuje také komunikaci přes rozhraní RS232 pomocí RS232 sériového portu a devíti vývody. Dále je na přípravku konektor pro připojení klávesnice nebo myši s rozhraním PS2. Pro zobrazení jednoduchých výsledků má přípravek čtyři sedmisegmentové displeje. Přípravek lze ovládat osmi spínači a má vyvedeno i 8 indikačních LED diod. Další rozšíření přípravku je možné pomocí tří portů se čtyřiceti vývody. rozšiřujících konektorů. Podrobnější informace o přípravku Spartan 3 jsou uvedeny v [4].

XC2-XL

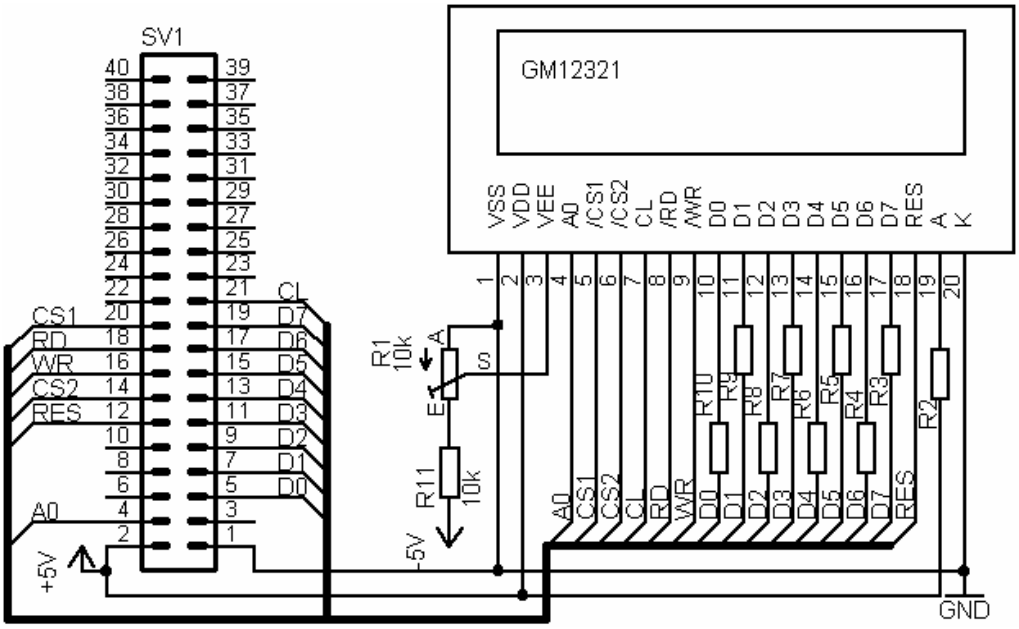
Tento vývojový kit obsahuje dva obvody CPLD. Jedním z nich je obvod CoolRunner-II XC2C256 v pouzdru TQ144 a druhým je XC9572XL v pouzdru VQ44. Vývojový kit má vyvedeny čtyři 40 pinové rozšiřující porty, kde jsou tři připojeny k obvodu XC2C256 a jeden k XC9572XL. Oba obvody lze programovat rozhraním JTAG. Taktování obou obvodů je zajištěno oscilátorem o frekvenci 1,8432 MHz, kde změnou taktovacího obvodu lze zvýšit frekvenci taktování až na 100 MHz. Ke každému obvodu je připojeno tlačítko, umístěné na desce kitu, které lze libovolně použít. Dále každý obvod ovládá jednu LED diodu, která je taky SW dostupná.

Obvod XC2C256 je navržen pro napájecí napětí 1,8 V. Obsahuje celkem 256 makrobuněk a v pouzdru TQ144 má 118 univerzálních vývodů. Výstupní piny lze nakonfigurovat do několika módů. Pro připojení displeje připadá v úvahu pouze mód LVCMOS33 nebo LVTTL. Další informace o celému přípravku lze nalézt v [5].

3.2. Napojení LCD displeje k vývojovému kitu

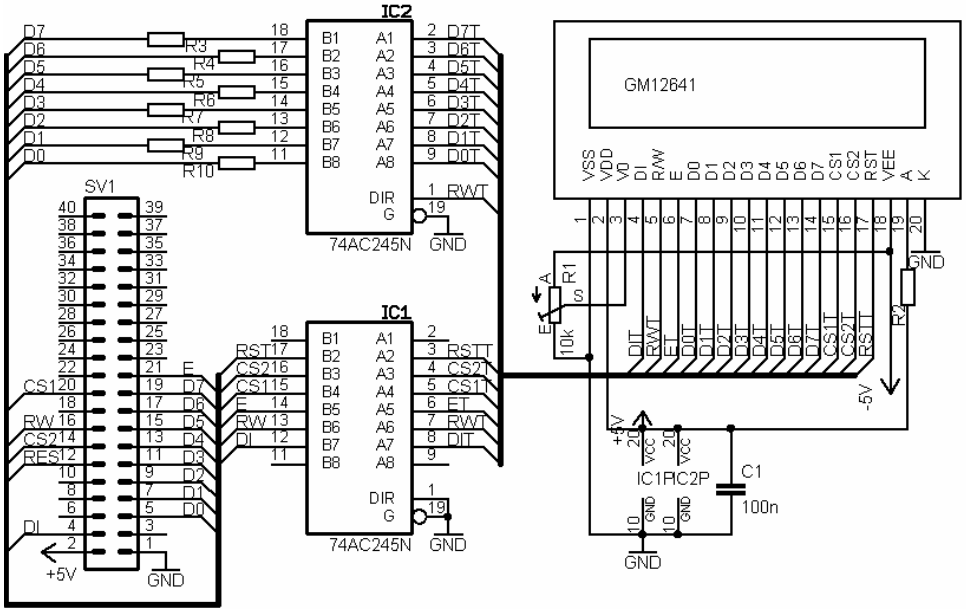
LCD displej je možné připojit k vývojovému kitu pouze pomocí rozšiřujících konektorů. Napěťové úrovně řadiče LCD displejů jsou 0 V nebo 5 V. Vývojový kit s CPLD nebo FPGA nepracuje s rovní 5 V, ale může se nastavit režim výstupu LVCMOS33, který změní vstupní i výstupní úrovně na 3,3 V. S tím mohou být problémy, kdyby použitý LCD displej měl rozhodovací úrovně jako obvody CMOS. Pak by 3,3 V považoval za neurčitý stav a displej by se nemohl ovládat. Displej s řadičem SED1520 má rozhodovací úrovně jako TTL logické obvody. Zbylé dva řadiče mají rozhodovací úrovně jako CMOS obvody. Proto je jednodušší připojit k programovatelnému obvodu displej GM12321 s řadičem SED1520.

Navrhované zapojení na Obr. 7 má připojeny ochranné odpory pouze u vývodů D0 až D7, protože zbylé vývody jsou vždy vstupy.



Obr. 7: Navrhované schéma zapojení

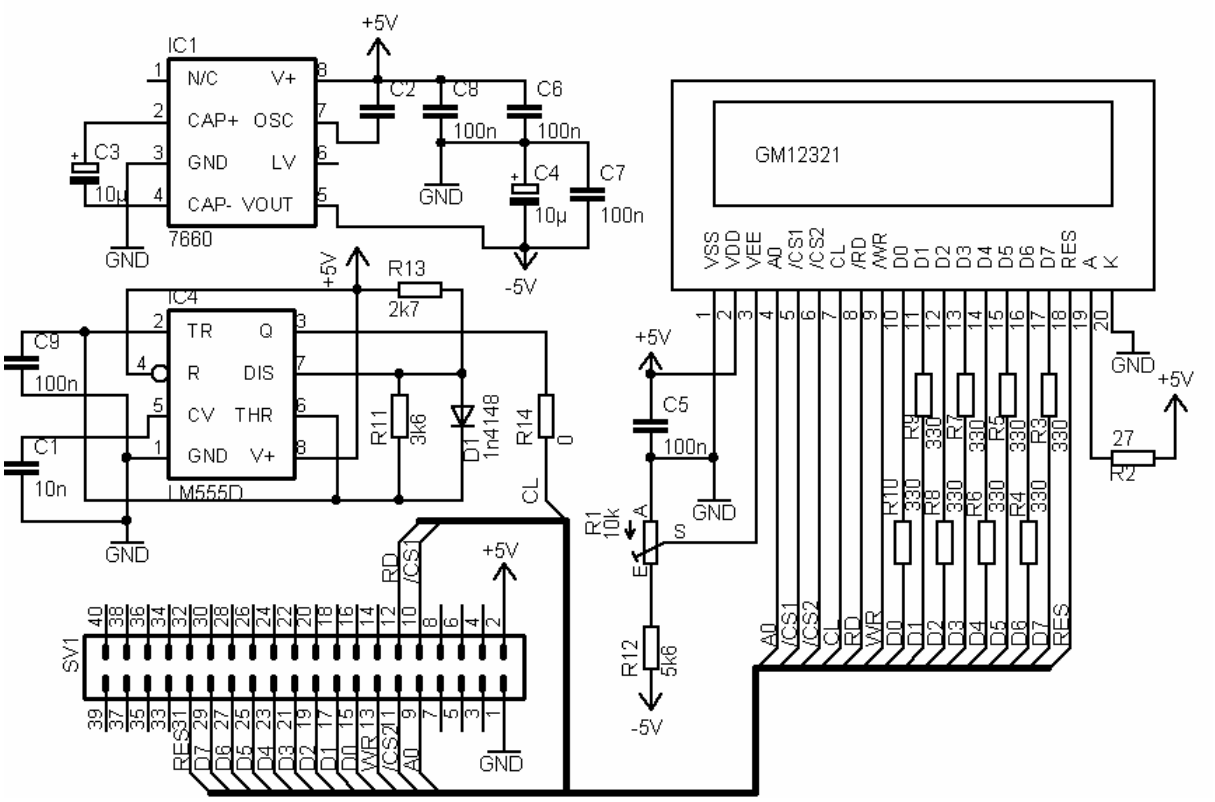
Pokud by bylo potřeba připojit displej, který má rozhodovací úrovně jako CMOS obvody, musel by se připojit mezi vývojový kit a LCD displej napěťový převodník. Použitelný napěťový převodník by byl 74HCT245, který je napájen z 5 V a má rozhodovací úrovně jako TTL logické obvody. Problém by mohl nastat při obousměrné komunikaci, protože obvod umožňuje přenos pouze jedním nebo druhým směrem. Díky vstupu displeje R/W, kterým se přečtou data z displeje, lze řídit i nastavení převodníku. Na Obr. 8 je schéma zapojení navrhovaného obvodu s převodníky napětí pro připojení LCD displeje s řadičem KS0107. Obvod IC1 je stále nastaven tak, aby byly výstupy aktivní a směr toku dat je od vývojového kitu k displeji. Obvod IC2 slouží pro oddělení datové sběrnice. Směr dat se nastavuje podle úrovně signálu R/W, kterým se konfiguruje displej.



Obr. 8: Navrhované schéma zapojení s napěťovým převodníkem

3.3. Kompletní návrh přípravku

Byl vybrán displej GM12321, který obsahuje řadič SED1520. V celkovém schématu je přidán zdroj záporného napětí, tvořený obvodem 7660, který plně vyhovuje pro napájení odporového děliče pro nastavení kontrastu. Dále displej GM12321 vyžaduje externí oscilátor 2 kHz. Oscilátor je realizován obvodem LM555 a je jím realizován astabilní klopný obvod. Displej je připojen k jednomu rozšiřujícímu portu, který má 40 pinů. Zapojení vývodů rozšiřujících portů je v [12]. Na Obr. 9 je celkové navrhované schéma modulu.



Obr. 9: Celkové navržené schéma zapojení

Kmitočet oscilátoru je dán odpory R11 a R13 a kondenzátorem C9. Pro přiblížení se střídě 1:1 byla do zapojení dodána dioda D1. Výpočet kmitočtu je dán vztahem (1), kde t1 a t2 jsou dány vztahem (2) a (3). Tyto vztahy jsou odvozeny v [13].

$$f = \frac{1}{t1 + t2}, \text{ [Hz; s, s]} \tag{1}$$

$$t1 = R11 * C9 * \ln(2), \text{ [s; Ω, F]} \tag{2}$$

$$t2 = R13 * C9 * \ln\left(\frac{2 * Ucc - 3 * Ud}{Ucc - 3 * Ud}\right), \text{ [s; Ω, F, V, V]} \tag{3}$$

Do vztahu (3) bylo za Ud zvoleno 0.6 V a Ucc 5 V. Hodnoty odporů byly vybrány z řady E24. Odpor R13 byl zvolen 2,7 kΩ, R11 dopočítán a jeho hodnota je 3,6 kΩ. Kapacita kondenzátoru C9 byla zvolena 100 nF.

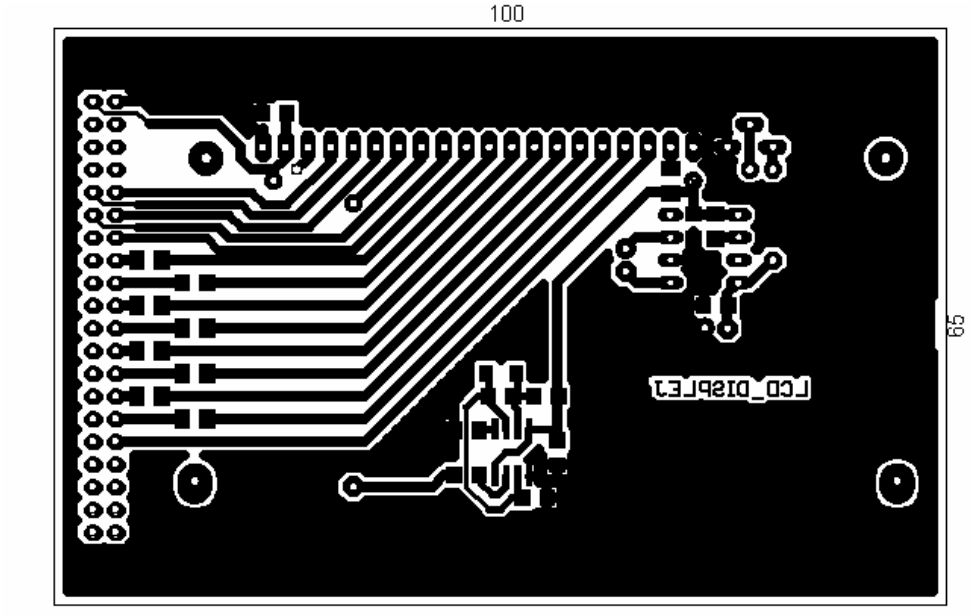
Ochranné odpory R3 až R10 byly zvoleny 330 Ω. Maximální proud z pinu displeje je daný vztahem (4a) při nekonfliktních úrovních a vztahem (4b) úrovňových kolizích.

$$I_{\max} = \frac{UCC - 3{,}3\,V}{330\,\Omega} = \frac{5 - 3{,}3}{330} \approx 5\,mA\,,\ [\text{A; V, }\Omega] \tag{4a}$$

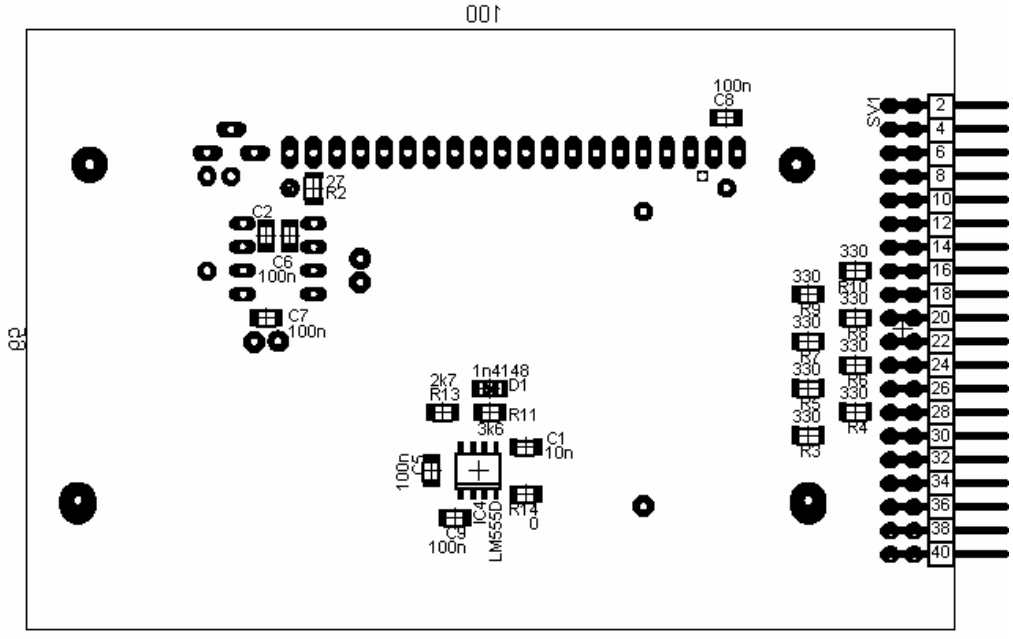
$$I_{\max} = \frac{UCC}{330\,\Omega} = \frac{5}{330} \approx 15\,mA\,,\ [\text{A; V, }\Omega] \tag{4b}$$

3.4. Návrh plošného spoje

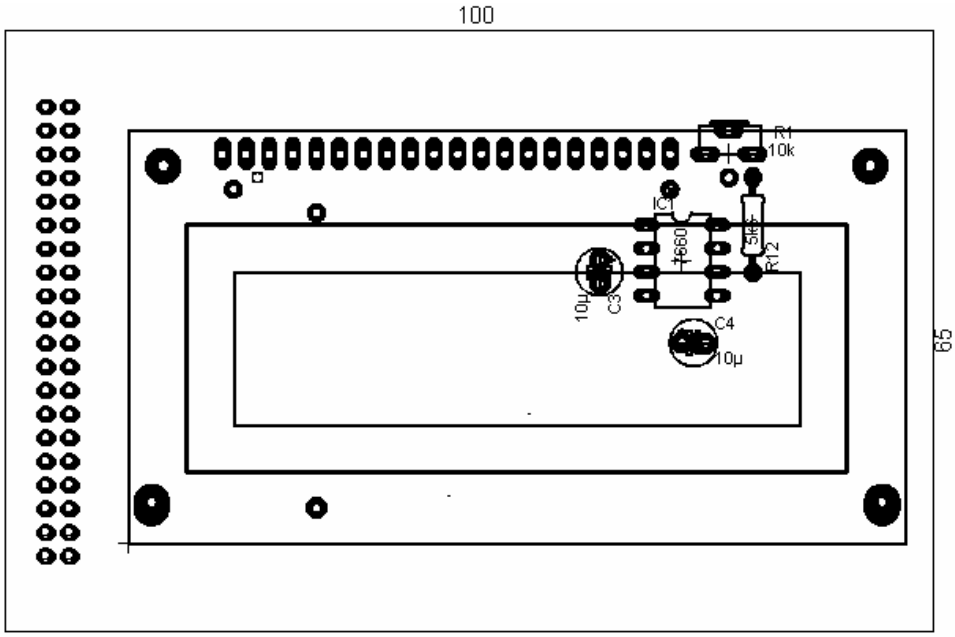
Navržený plošný spoj má rozměry 100x65 mm a je na Obr. 10 s předlohou pro osazení na straně plošného spoje na Obr. 11 a s předlohou osazení na straně součástek na Obr. 12.



Obr. 10: Navržená deska plošného spoje



Obr. 11: Předloha pro osazení součástek ze strany plošného spoje



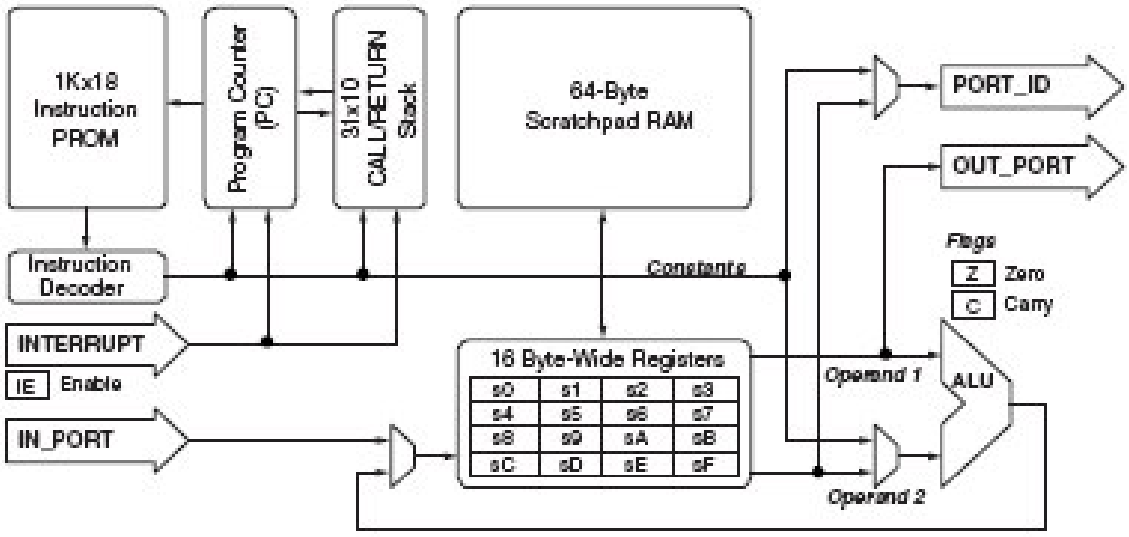
Obr. 12: Předloha pro osazení součástek

4. Blok řízení displeje s procesorem PicoBlaze

Základním úkolem bloku řízení je propojení pinů externího portu, kam je připojen displej, s procesorem PicoBlaze. Předpokládá se připojení desky displeje na port B1 vývojového kitu Spartan 3. Deska displeje nevyužívá všechny piny portu B1. Využité piny jsou dány návrhem desky v kapitole 3. Dále tento blok řízení vytváří obousměrnou datovou a řídící sběrnici připojenou k displeji. Tento blok řízení bude obsahovat rozhraní mezi procesorem PicoBlaze a okolím, které se bude monitorovat. Protože displej GM12321 nemá font symbolů, je nutné v této části také připojit k procesoru paměť ROM, ve které budou definovány jednotlivé symboly.

4.1. Procesor PicoBlaze

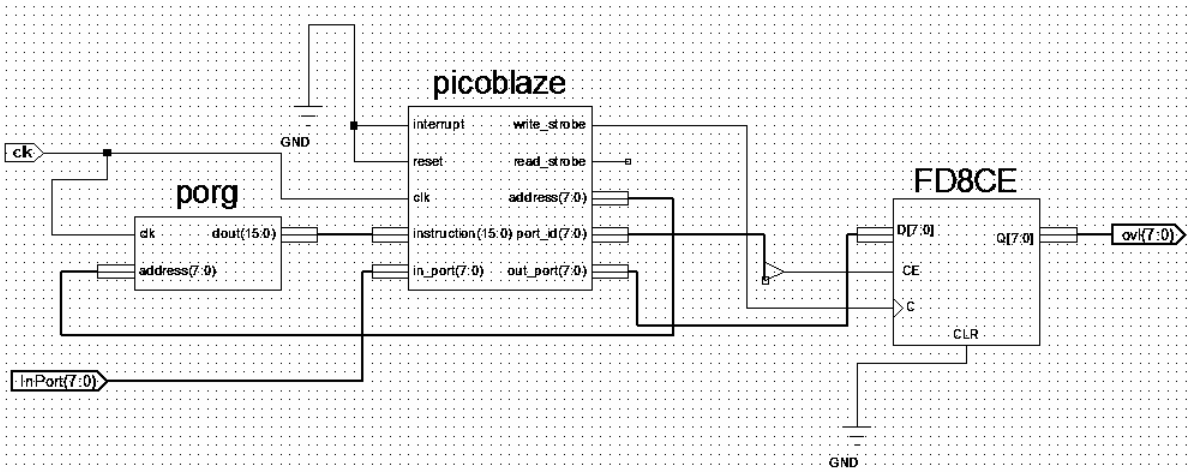
Jedná se o procesor osmibitový s 16 univerzálními datovými registry. Velikost programové paměti může být až 1000 instrukcí pro obvody FPGA a 256 instrukcí u CPLD obvodů. Obsahuje 8 bitovou aritmetickou jednotku s příznaky ZERO a CARRY. Dále obsahuje 64 B interní paměti RAM. Umožňuje adresovat až 256 okolních osmibitových zařízení, vstupních i výstupních. Stack pointer dovoluje volat až 31 podprogramů najednou. Tento procesor při taktovací frekvenci 200 MHz v obvodu FPGA dosahuje výkonnosti 100 MIPS. Odezva na přerušení není delší než 5 taktů oscilátoru. Základní blokové schéma procesoru je na Obr. 13.



Obr. 13: Blokové schéma procesoru PicoBlaze

Implemetace procesoru PicoBlaze do CPLD

Procesor PicoBlaze se skládá z několika modulů napsaných v jazyku VHDL. Všechny moduly spojuje soubor PicoBlaze.vhd. Po načtení hierarchie celého procesoru lze vygenerovat schématickou značku. Tato schématická značka má vyvedenu adresní a datovou sběrnici pro připojení paměti programu. Paměť programu se vygeneruje programem asm.exe, který byl stažen spolu s moduly procesoru PicoBlaze. Tato paměť je v kódu VHDL. Na Obr. 14 je znázorněn procesor PicoBlaze pro obvody CPLD s rozšířením pro jednoduché testování.



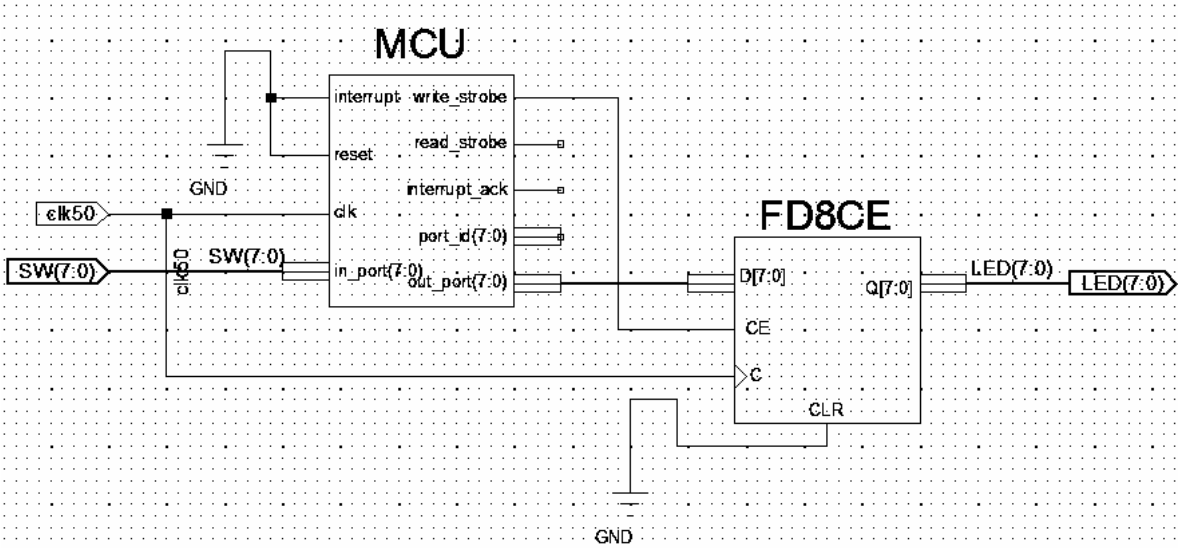
Obr. 14: Schéma procesoru PicoBlaze pro otestování funkce v obvodu CPLD

Procesor má dále vyveden taktovací vstup, resetovací vstup, vstup pro volání externího přerušení, vstupní port, potvrzovací signál externího přerušení a bránu PORT_ID, která slouží pro určení adresy portu.

Díky jednoduchému programu bylo ověřeno, že procesor PicoBlaze byl do obvodu CPLD přeložen dobře a chová se podle zadaného programu. Program měnil rychlost blikání LED diody podle stavu pomocného tlačítka

Implemetace procesoru PicoBlaze do vývojového kitu Spartan 3

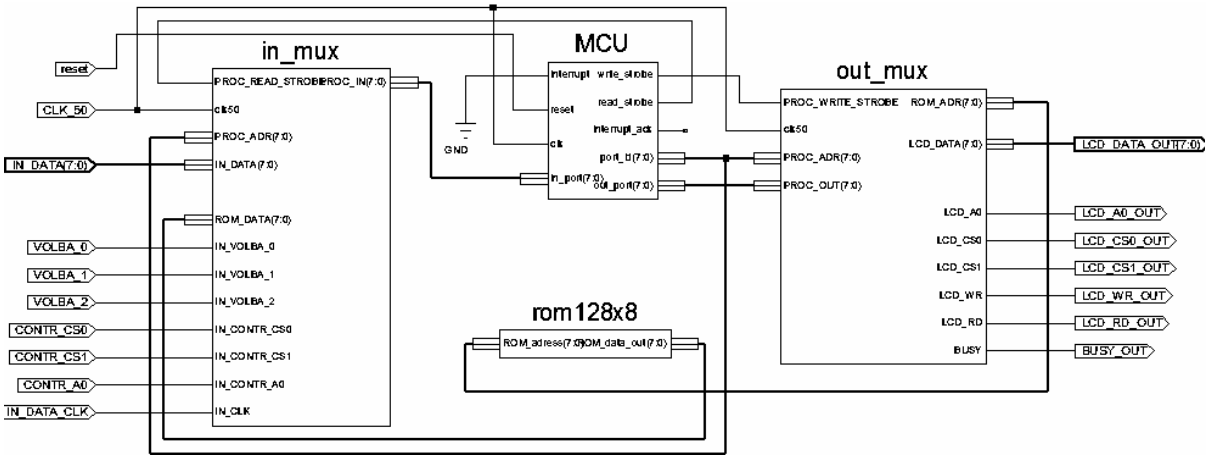
Procesor PicoBlaze pro obvody FPGA má odlišný zdrojový kód. Na stránkách firmy Xilinx lze získat zdrojový kód procesoru, který už v sobě obsahuje paměť programu. Zdrojové soubory mají označení kcpsm3.vhd a MCU.vhd a soubor s pamětí programu má označení prog_ROM.vhd. Na Obr. 15 je znázorněno schéma pro jednoduché otestování procesoru v obvodu FPGA.



Obr. 15: Schéma procesoru PicoBlaze pro otestování funkce v obvodu FPGA

4.2. Schéma struktury propojení procesoru s displejem

Pro základní ovládání LCD displeje s řadičem SED1520 se vyžaduje pouze zápis dat z datové sběrnice řídícími signály. Z displeje lze číst aktuální zobrazované body, ale i registr, ve kterém jsou informace o současném stavu. Pro jednoduché ovládání není nutné číst z displeje data. Tím se velmi zjednoduší blok řízení, protože mezi procesorem a displejem bude jen multiplexer s výstupní pamětí. Schéma jednoduché struktury propojení procesoru s displejem je na Obr. 16.



Obr. 16: Schéma napojení procesoru PicoBlaze k displeji

Celý návrh byl pojat jako ovladač s jednoduchým nastavením a minimální nutnou obsluhou. Program v procesoru PicoBlaze podle stavů vstupů VOLBA_0 až VOLBA_2 zpracuje vstupní data IN_DATA a řídící signály CONT_CS0, CONTR_CS1 a CONTR_A0 při vzestupné hraně vstupního signálu IN_DATA_CLK. Po přijetí a zahájení zpracování je programem nastaven signál BUSY_OUT. Tímto signálem se říká předchozímu bloku, že další zapsaná data budou ignorována a předchozí blok musí počkat na jeho sestupnou hranu. LCD displej je připojen k procesoru datovou sběrnicí LCD_DATA_OUT a řídícími signály LCD_A0_OUT, LCD_CS0_OUT, LCD_CS1_OUT, LCD_WR_OUT a LCD_RD_OUT.

Implementované funkce

Použitý grafický LCD displej GM12321 vyžaduje dodržet časovou posloupnost přivedených dat tak, aby displej tyto data spolehlivě a správně přečetl. Protože dodržení časové posloupnosti přivedených dat by zbytečně zvyšovalo nároky na zdroj dat, byly do algoritmu v procesoru zahrnuty funkce pro zápis dat dle doporučení v katalogovém listu displeje [9]. Dále LCD displej rozlišuje 2 typy dat. Jedním typem jsou konfigurační a druhým jsou data určená pro zobrazení. Rozlišováním typů dat by opět zbytečně zvyšovalo nároky na zdroj dat, a proto i toto rozlišení dat zajistí algoritmus v procesoru.

LCD displej je rozdělen do dvou stejných částí. Volba částí je zajištěna vstupy CS0 a CS1. Konfigurací displeje lze nastavit libovolný řádek a libovolný sloupec na který se zapíší data. Displej má funkci automatického posouvání sloupce. Díky této funkci není nutné zadávat číslo nového sloupce po každém odeslání dat pro zobrazení a hodnota sloupcového registru se automaticky zvýší o 1. Při odesílání souvislých dat se musí ošetřit přechody mezi jednotlivými částmi, případně automaticky přejít na nový řádek. Z toho vyplývá, že algoritmus musí uchovávat informace o aktuálním řádku a sloupci, aby bylo možné určit, kdy je potřeba odesílat data druhé části displeje.

Algoritmus by bylo dobré vybavit i možností nastavení nového řádku, případně nastavení počátku displeje, aby zdroj dat dokázal jednodušeji obsluhovat displej. Tyto funkce ukládají nastavení přímo do paměti displeje a současně nastaví hodnoty řádku a sloupce v algoritmu, aby algoritmus měl správná data a přepínal mezi částmi displeje podle skutečného stavu displeje.

Displej nemá zabudovanou žádnou tabulku znaků. Příchozí data, určená pro zobrazení se na displeji objeví v podobě sloupce vysokého 8 pixelů. Pokud aplikace vyžaduje zobrazit symboly, musela by převést číslo symbolu do šestice po sobě jdoucích dat, které vyobrazí požadovaný znak. Tento požadavek by zvyšoval náročnost obsluhy displeje, proto by ovladač displeje měl umět převést vstupní číslo na znak. Aby zdroj dat mohl stále displej nakonfigurovat jinak, než je přednastaveno, měl by umět předat všechny řídící a datové signály v nezměněné podobě displeji.

Z předchozího rozboru je patrné, že pro snadné obsluhování displeje je potřeba, aby algoritmus měl funkci nulování, nastavení dalšího řádku, nastavení libovolného sloupce a řádku, automatický posuvník, vkládání znaků z paměti ROM a přímé vkládání dat do paměti displeje.

Realizace IN_MUX

Procesor PicoBlaze má 8 bitový vstupní port s možností volby až 256 adres daných adresovým portem. Blok IN_MUX je brána mezi vstupem procesoru a okolím. Podle požadovaných funkcí celého bloku řízení displeje jsou vyžadovány tři 8 bitové porty. Blok IN_MUX je 8 bitový registrový multiplexer, který zachytává data při náběžné hraně vstupu PROC_READ_STROBE. První port jsou vstupní data, druhý port jsou data z paměti ROM a třetí port jsou řídící signály IN_VOLBA_0, IN_VOLBA_1, IN_VOLBA_2, IN_CONTR_CS0, IN_CONTR_CS1, IN_CONTR_A0 a IN_CLK. Na výstup bloku se přepnou data podle aktuálního stavu PROC_ADR. Protože je šířka adresové sběrnice 8 bitů, přísluší pro zjednodušení každému vstupnímu portu 1 vodič sběrnice.

Realizace OUT_MUX

Použitý procesor má 8 bitový výstupní port. Také lze volit až 256 adres při zápisu na port. Blok OUT_MUX zajistí zapamatování výstupního portu při dané adrese. Požadované funkce vyžadují minimálně 3 porty. První port je pro výstup dat k displeji LCD_DATA, druhý port ROM_ADR generuje adresu pro paměť ROM a na třetím portu jsou řídící signály pro displej LCD_A0, LCD_CS0, LCD_CS1, LCD_WR, LCD_RD a také řídící výstup BUSY_OUT, který dává informaci předchozímu bloku, že ovladač displeje pracuje a nemůže přijímat další data.

Paměť ROM s fontem symbolů

V návrhu byla použita šablona ze systému ISE pro snadné vytvoření paměti ROM s požadovanou velikostí. V [14] je uvedeno, kde lze získat šablonu.

Šířka adresní sběrnice udává maximální počet symbolů v paměti. Pro ukázku byl vytvořen číselný font s písmeny A až F. Aby se mohl navrhovaný algoritmus programu co nejvíce zjednodušit, bylo přiděleno jednomu symbolu 8 adresových míst. Celý znak vyžaduje jen 6 paměťových míst. Sice nejsou využity 2 Byte v paměti na každý symbol a velikost paměti je proto o třetinu větší, ale tento systém umožňuje rychle oddělit adresu znaku od adresy jednotlivých sloupců znaku. Tato velká výhoda zastíní mírné zvýšení

velikosti ROM. Pokud by byl vyžadován větší znakový font, je nutné rozšířit adresní sběrnici. To by znamenalo, že by modul OUT_MUX musel být rozšířen o další port, který by udával vyšší byte adresy.

Implementace do obvodu CPLD

Pro implementaci procesoru PicoBlaze do obvodu CPLD bylo navrženo schéma zapojení, na kterém by se ověřila funkce navrženého programu. Během překladu kódu pro obvod XC2C256 došlo k chybě, protože tento obvod je malý a proto nemohlo být ověřeno navrhované schéma s programem.

4.3. Program pro procesor PicoBlaze

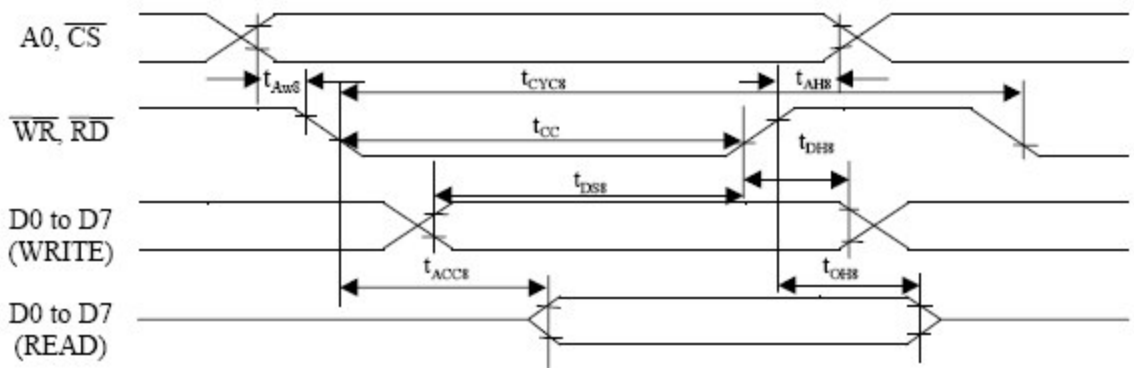
Procesor PicoBlaze podle navrhovaného schéma zapojení má připojen displej na dvou osmibitových portech. Adresa datového vstupu displeje je 1 a adresa ovládacího registru je 2. Adresa pro paměť ROM je 4. Díky tomuto adresování se určí, do kterého výstupu se pošlou data. Pomocný signál BUSY-OUT nesouvisí s řídícími signály pro LCD displej, ale protože je zbytečné pro 1 bitový signál rezervovat další celý port, byl tento signál přidán k řídícím signálům LCD displeje. Na to je třeba pamatovat při jakékoliv změně řídících signálů LCD displeje.

Funkce zápisu dat

Výrobce displeje uvádí, že se displej může připojit ke sběrnici procesoru řady 80 nebo ke sběrnici procesoru řady 68. Pro každý typ sběrnice platí jiná pravidla komunikace. Volba sběrnice se provádí nastavením jednoho vstupu sběrnice. Byla zvolena sběrnice pro procesor řady 80.

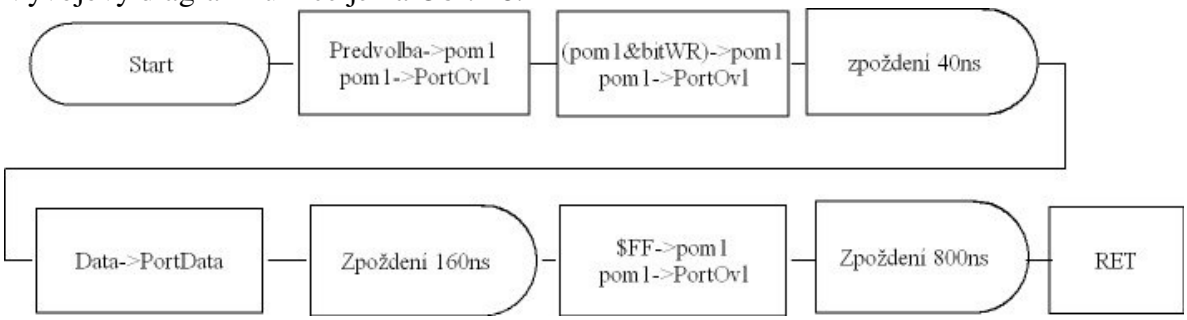
Sběrnice pro procesory řady 80 obsahuje šestnáctibitovou adresní sběrnici a osmibitovou datovou sběrnici. Dále je řídící sběrnice tvořena signály /WR, /RD a ALE. K displeji je připojena datová sběrnice, signály /RD a /WR. Z adresní sběrnice se pomocí adresového dekodéru musí generovat signály CS1, CS2 a A0, kterými se vybírá, která část displeje se bude měnit a jestli se budou odesílat displeji data nebo příkaz. Podrobnější informace jsou uvedeny v [9].

Displej v módu sběrnice pro procesory řady 80 potřebuje před zahájením přenosu nastavit vstupy A0 a CS. Dále se musí nastavit vstup pro čtení nebo zápis přes vývody RD a WR. Aby displej byl správně nastaven, musí se po nastavení A0 a CS počkat 20 ns, než se může nastavit signál pro zápis nebo čtení. Platná data musí být nastavena na datové sběrnici minimálně 80 ns před náběžnou hranou a 10 ns po náběžné hraně signálu RD. Tento signál musí být v úrovni L po dobu 200 ns. Po náběžné hraně signálu RD se může změnit předvolba vývodů A0 a CS, ale nemůžou se zapisovat další data. Další data se můžou zapsat až za 800 ns. V této době se nepoužívá sběrnice a proto lze zapisovat do druhé části displeje. Na Obr. 17 je graficky znázorněna posloupnost řídících signálů při ukládání dat.



Obr. 17: Časový diagram řídících signálů při zápisu i čtení dat

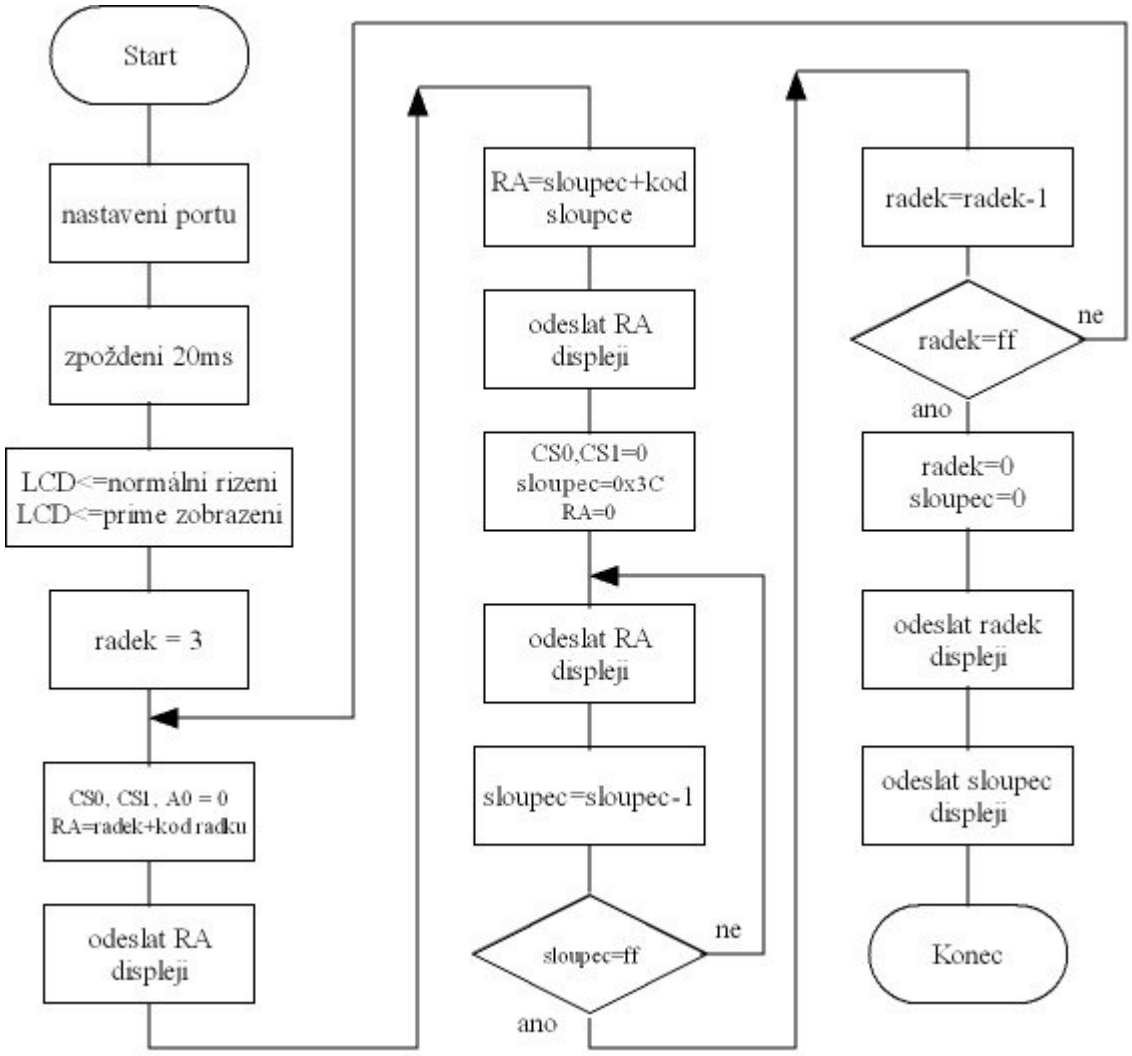
Funkce pro zápis musí mít zadaná data, která se budou přenášet a ukládat v displeji a nastavení řídících signálů A0, CS0 a CS1. Data, která se uloží v displeji se musí uložit před voláním funkce do registru Data. V registru Predvolba musí být nastaveny bity signálů A0, CS0 a CS1. Tímto registrem se určuje, zda se bude displej nastavovat a posílat příkaz nebo se data zapíší do paměti RAM displeje. V této funkci musí být realizováno zpoždění. Taktovací frekvence procesoru je ve vývojovém kitu Spartan 3 je 50 MHz. Vykonání jedné instrukce vyžaduje 2 strojové cykly. Potom vykonání jedné instrukce trvá 40 ns. Proto pro zpoždění 10 ns nebo 20 ns nebude potřeba realizovat funkci zpoždění. Zpoždění 200 ns je vytvořeno několika instrukcemi Nop. Zpoždění 800 ns by vyžadovalo příliš mnoho instrukcí Nop, proto je vytvořena smyčka, kterou se vícekrát provede stejný úsek programu. Vývojový diagram funkce je na Obr. 18.



Obr. 18: Vývojový diagram funkce pro zápis dat

Inicializace displeje

Displej GM12321 nevyžaduje složitou inicializaci. Pokud není potřeba jiné nastavení, než je automaticky nastaveno po zapnutí napájení, pak stačí displej pouze smazat a příkazem aktivovat zobrazování. Displej je nutné smazat, protože se obsah interní paměti automaticky nenuluje po připojení napájení. Pokud byl displej dlouho vypnutý, po jeho zapnutí je obsah displeje vynulován a nesvítí žádné pixely. Pokud bylo přerušeno napájení jen krátce, obsah na displeji je nedefinovaný a proto je nutné ho po každém zapnutí smazat. Vývojový diagram funkce pro inicializaci displeje je na Obr. 19.



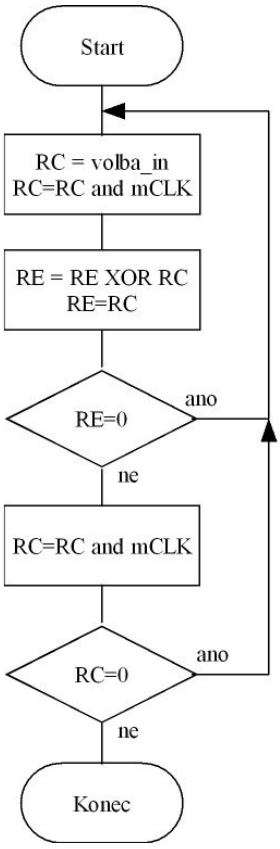
Obr. 19: Vývojový diagram funkce pro inicializaci displeje

Detekce náběžné hrany

Jedním z důležitých vstupních signálů je taktovací vstup. Tímto signálem je procesoru sděleno, že na jeho vstupech jsou platná data, okamžitě zahájí svou činnost a podle aktuálního stavu vstupů předá vstupní data LCD displeji. Doba reakce procesoru přímo závisí na způsobu provedení algoritmu pro detekci impulzu. Aby se zabránilo opakovanému vykonávání příkazu, kdyby zdroj dat měl pomalé reakce a nechal aktivující úroveň delší dobu než ovladač potřebuje k předání dat, je vhodné, aby procesor reagoval pouze na změnu úrovně taktovacího pulzu.

Byl zvolen algoritmus detekce hrany s využitím instrukce XOR. Díky této instrukci lze určit, jestli jsou bity ve dvou registrech shodné nebo rozdílné. Když se budou porovnávat 2 bity, kde jeden bude aktuální a druhý jeden takt zpožděný, potom se touto instrukcí detekují náběžné a sestupné hrany. Proto po detekci změny úrovně se musí provést druhý test, kterým se zjistí, jestli šlo o vzestupnou nebo sestupnou hranu.

Je možné detekovat hranu i dalším způsobem. Testovalo by se, jestli zpožděný bit je nula a aktuální je jedna. Výsledný algoritmus by buď vyšel stejně jak algoritmus s funkcí XOR nebo by vyšel s rychlejší reakcí, ale muselo by se předpokládat, že se nezmění hodnota registru během vykonávání příkazu a to by mohlo vytvářet záludné chyby. Proto je v návrhu algoritmus s instrukcí XOR. Na Obr. 20 je vývojový diagram detektoru vzestupné hrany.

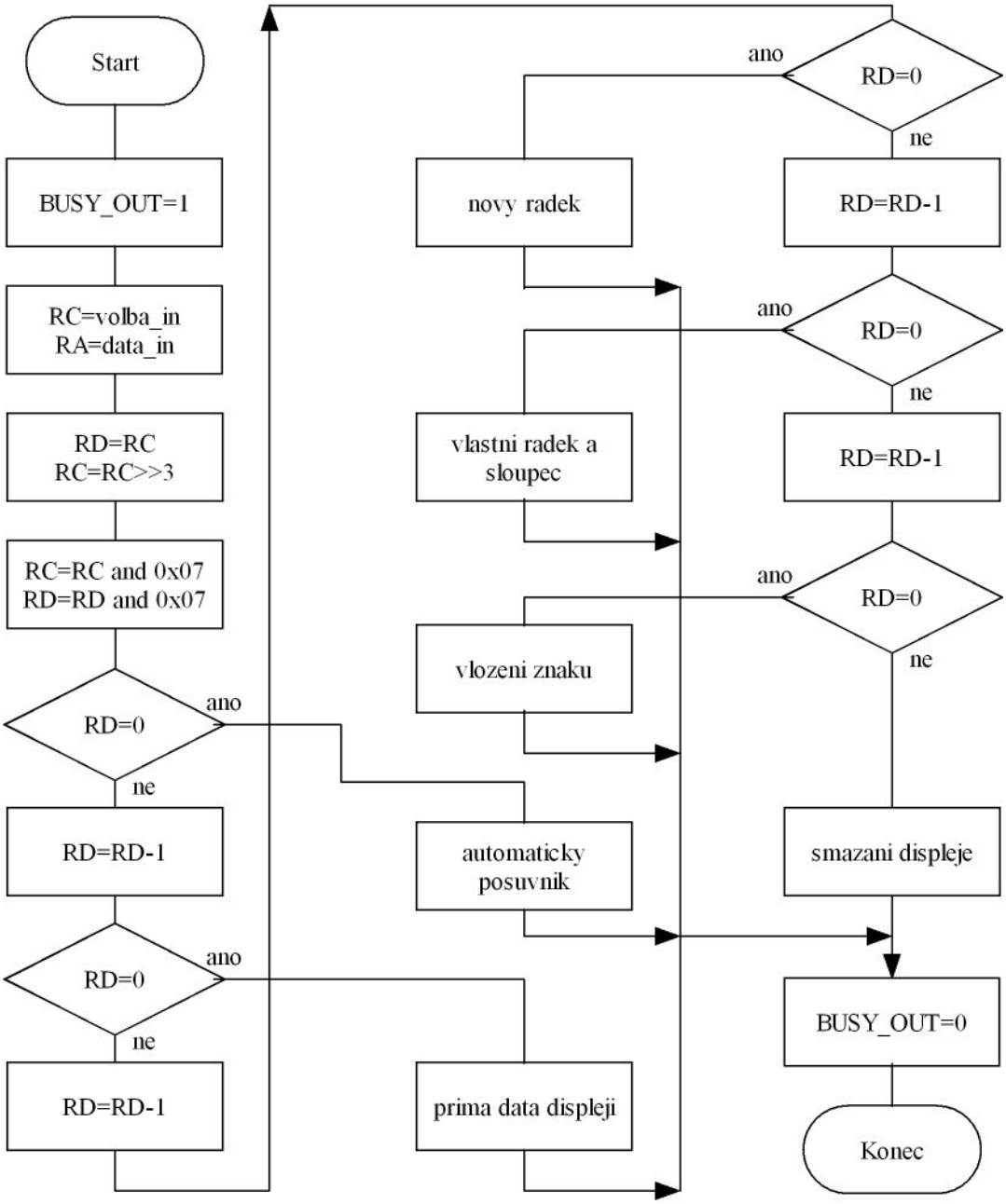


Obr. 20: Vývojový diagram funkce pro detekci náběžné hrany

Jedná se o blokující funkci, která přestane blokovat procesor, jakmile detekuje náběžnou hranu. Ve své blokující smyčce cyklicky čte ze vstupu, kam je připojen vstupní taktovací signál. Perioda vzorkování je 5 instrukcí, při sestupné hraně proběhne 7 instrukcí do dalšího čtení. Ve vývojovém diagramu je malá nepřesnost. Test registru RE je proveden funkcí XOR a vyhodnocen až za zkopírováním obsahu RC do RE. Logický součin registru RC a masky taktovacího pulzu se provádí pouze pro generování příznakových bitu pro následující rozhodování.

Zpracování příkazů

Jakmile je detekována vzestupná hrana taktovacího pulzu, znamená to, že na vstupech je platná předvolba a jsou platná data. Tento blok programu zpracuje a vyhodnotí příkaz, který je volen vstupními signály IN_VOLBA_0, IN_VOLBA_1, IN_VOLBA_2. Je nutné, aby program v co nejkratším čase uchoval stav vstupních dat. Tím se zkracuje nutná doba udržení předvolby od zdroje dat. Protože vstupy volby příkazu a kontrolní vstupy pro přímé ovládání displeje IN_CONTR_CS0, IN_CONTR_CS1, IN_CONTR_A0 jsou sdruženy na jednom portu, je nutné tyto nesouvisející vstupy oddělit. Nejsnáze lze toho docílit vhodnou maskou s následnou rotací registru. Když jsou kód volby a kontrolní vstupy odděleny, následuje rozvětvení algoritmu na zpracování a předání dat displeji podle hodnoty kódu volba. Vývojový diagram bloku je na Obr. 21.

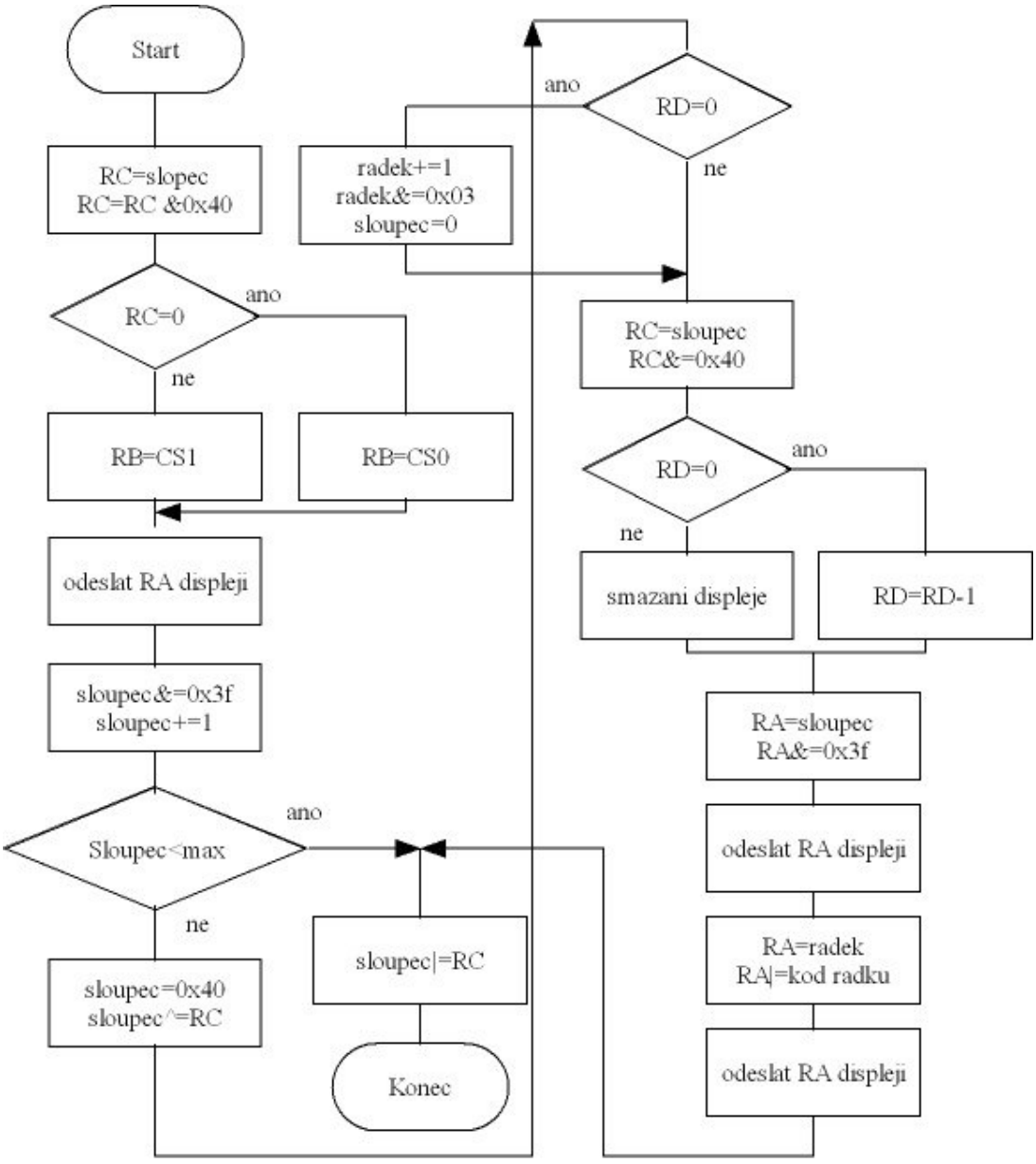


Obr. 21: Vývojový diagram funkce pro zpracování a vykonání příkazu

Funkce automatický posuvník

Tato funkce zajistí plynulý pohyb vkládaných dat po celém displeji. Při této volbě procesor automaticky přepíná části displeje podle aktuální polohy kurzoru a po posunutí kurzoru na konec řádku se automaticky posune na nový řádek. Aby procesor dokázal rozpoznat, kdy je potřeba zapisovat do druhé části displeje a současně nemusel před každým zapsaným sloupcem číst z displeje aktuální číslo sloupce, je nutné, aby procesor uchoval tuto informaci v paměti. Tímto bohužel vzniká zdroj možný problémů. Kdyby zdroj dat nevyužil předpřipravené funkce na změnu polohy kurzoru a změnil polohu přímým zapsáním displeji, program v procesoru by předpokládal jiný aktuální sloupec a řádek a došlo by k chybnému přechodu z jedné části displeje na druhou. Správným ovládáním ovladače displeje k tomu nemůže dojít, proto výhody tohoto řešení převažují. Tento jediný problém by se dal řešit. Procesor by odposlouchával, co se posílá displeji a potřebné informace by odchytil a uložil.

V případě potřeby je možné tuto funkci v programu realizovat. Vývojový diagram automatického posuvníku je na Obr. 22.



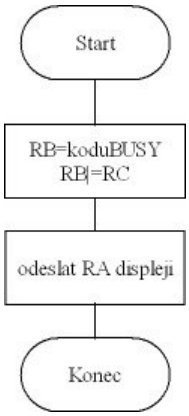
Obr. 22: Vývojový diagram automatického posuvníku

Algoritmus je koncipován tak, že LCD displej je nakonfigurován od předchozí komunikace a je možné hned zapsat data, která se zobrazí. Podle hodnoty aktuálního řádku se určí, do které části displeje se má zapisovat. Po zapsání dat se automaticky zvýší hodnota registru Sloupec o 1 a následně se provádí test, jestli další data by se nezapsala mimo rozsah displeje. Pokud hodnota sloupce nezasahuje mimo rozsah displeje, sloučí se aktuální pozice sloupce s příznakem rozlišujícím aktivní část displeje.

Při překročení rozsahu displeje se testuje, jestli je kurzor na konci řádku nebo v půlce. Pokud je na konci řádku, zvýší se registr Radek o 1 a registr Sloupec se vynuluje. Nové hodnoty registrů Sloupec a Radek se zapíší do jedné části displeje podle příznakového bitu v registru RC. Následuje sloučení příznakového bitu v RC s registrem Sloupec a opuštění funkce.

Funkce zápisu přímých dat

Přímá data se zapíší ze vstupů IN_CONTR_CS0, IN_CONTR_CS1, IN_CONTR_A0 a IN_DATA na výstupy LCD_A0, LCD_CS0, LCD_CS1 a LCD_DATA. Funkce dodrží veškeré časové zpoždění pro správné uložení dat. Vývojový diagram funkce je na Obr. 23.

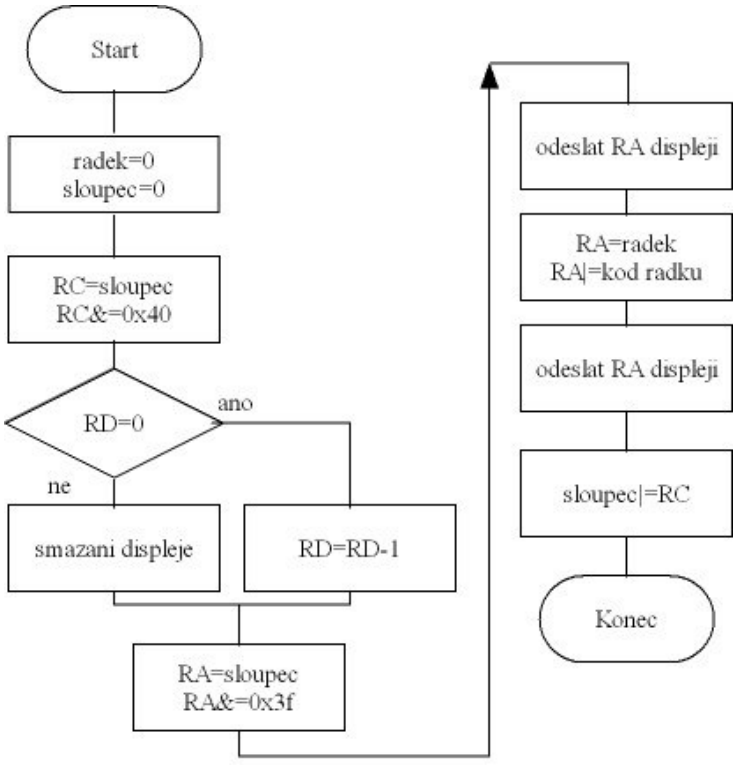


Obr. 23: Vývojový diagram algoritmu vkládání přímých dat

Algoritmus uloží do registru nutnou předvolbu pro zachování požadovaného stavu vývodů RB, WR a výstupu BUSY_OUT. Dále se provede logický součet registru RB s předvolbou RC. Na závěr se odešlou data displeji.

Funkce nulový řádek a nulový sloupec

Tato funkce vynuluje registry Radek a Sloupec a uloží nové hodnoty do displeje. Tím se dostane kurzor na počátek displeje, aniž by se změnil jeho obsah. Vývojový diagram této funkce je na Obr. 24.

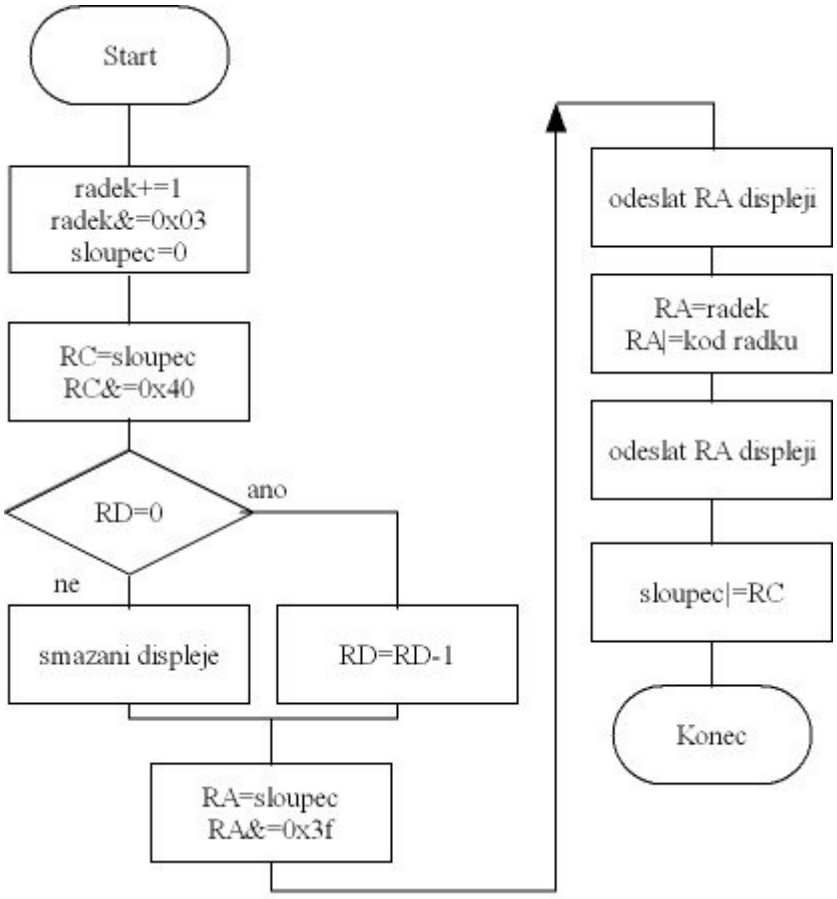


Obr. 24: Vývojový diagram algoritmu nulování sloupce a řádku

Ve funkci se v úvodu vynulují proměnné Radek a Sloupec a podle jejich hodnot se provede v displeji předvolba, podle které se následující data zapíšou na počátek displeje. Dále se oddělí od proměnné Sloupec příznak, který říká, do které části se má zapsat. Podle tohoto bitu se zvolí CS0 nebo CS1. V závěru je displej nastaven na počátek prvního řádku.

Funkce nový řádek

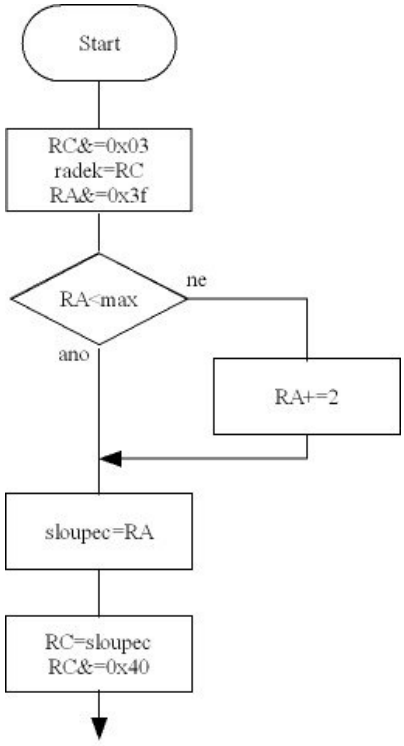
Tato funkce je velmi podobná té předchozí, protože po změně čísla řádku s maskováním 0x03, které zajistí při neustálém přičítání jedničky postupnou změnu čísel v intervalu 0 až 3, jsou nové hodnoty zapsány displeji. Vývojový diagram této funkce pro nastavení nového řádku je na Obr. 25.



Obr. 25: Vývojový diagram algoritmu nastavení nového řádku

Funkce uživatelský řádek uživatelský sloupec

Tatro funkce opět zjednodušuje vkládání dat do určitého místa na displeji. Zdroj dat může nastavit pozici, kam se mají následující data uložit. Informace o číslu sloupce se očekává na vstupu IN_DATA a řádek je uložen na vstupech IN_CONTR_CS0, IN_CONTR_CS1. Vývojový diagram je hodně podobný předchozím dvěma, rozdíl je jen v úvodním bloku, kde se nastavují hodnoty proměnných Sloupec a Radek. Číslo sloupce se musí upravit tak, aby číslo sloupce odpovídalo skutečnosti. Jedna polovina použitého displeje má 62 sloupců. Celý algoritmus předpokládá, že displej má 64 sloupců. Je proto nutné v případě překročení hodnoty 62 přičíst hodnotu 2, aby nastavený počet odpovídal skutečnosti. Na Obr. 26 je uveden začátek algoritmu funkce uživatelský řádek a sloupec.



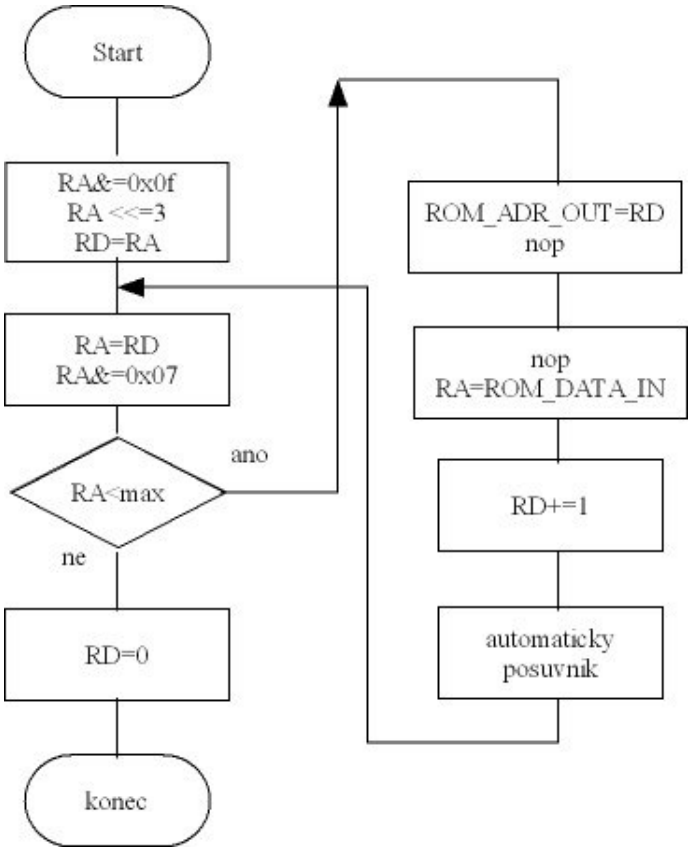
Obr. 26: Vývojový diagram začátku algoritmu funkce uživatelský řádek a sloupec

Vkládání znaku z paměti ROM

Všechny předchozí funkce vyžadovaly na jejich vykonání celkem krátkou dobu. Vždy šlo v ideálním případě jen o jednu komunikaci s displejem a v nejhorším případě bylo potřeba nastavit číslo nového řádku a sloupce. Celkem nebyly překročeny 3 komunikace s displejem. Tato funkce vyžaduje daleko větší počet komunikací s displejem. Tím je dána i delší doba vykonání. Při zápisu symbolu na displej by bylo výhodné, aby při nepřetržitém vkládání znaků zdroj dat nemusel kontrolovat přechody mezi částmi displeje a konce řádků. Funkce automatického posuvníku je celkem složitá, proto by v ideálním případě měl algoritmus využívat již hotové funkce. Na Obr. 27 je vývojový diagram funkce pro vyčítání znaku z paměti ROM.

Aby se mohla hotová funkce posuvníku využít, je nutné vložit na konec této funkce podmínku, kterou se zajistí vracení programu zpět na funkci pro vkládání znaku z paměti ROM. Registr RD byl využit jako počítadlo již vyčtených a uložených sloupců znaku, proto lze jeho testováním rozlišit, jestli má program pokračovat ve funkci pro vkládání znaku nebo přejít na detekci vstupního pulzu. Registr RD je už využit při určení vybraného příkazu. Způsob určení příkazu zajistí jeho vynulování, proto nemůže dojít k chybnému vyhodnocení při rozhodování ve funkci posuvníku. Z toho vyplývá, že při nulové hodnotě registru RD ve funkci posuvníku se musí funkce ukončit a přejít na detekci taktovacího pulzu.

Pro ukázku algoritmu pro vkládání symbolů byl vytvořen znakový font čísel s písmeny A až F pro vytvoření HEX čísel. Takový font vyžaduje celkem 16 symbolů. Pro každý symbol je nutné rezervovat 6 Byte. To by znamenalo velikost paměti ROM o velikosti 16*6 = 96 Byte. U takto organizované paměti by znamenalo při adresování vrcholu každého symbolu násobit kód znaku šesti. V případě, že by pro každý znak bylo vyhrazeno 8 Byte, potom by bylo nutné pro vygenerování vrcholové adresy symbolu násobit kód znaku osmi a to lze ve dvojkové soustavě realizovat trojitou rotací. Trojitá rotace je s procesorem daleko snazší než násobení kódu znaku šesti. Proto bylo v návrhu rezervováno pro každý symbol 8 Byte. Z toho plyne, že velikost paměti ROM pro tuto znakovou sadu bude 16*8 = 128 Byte.



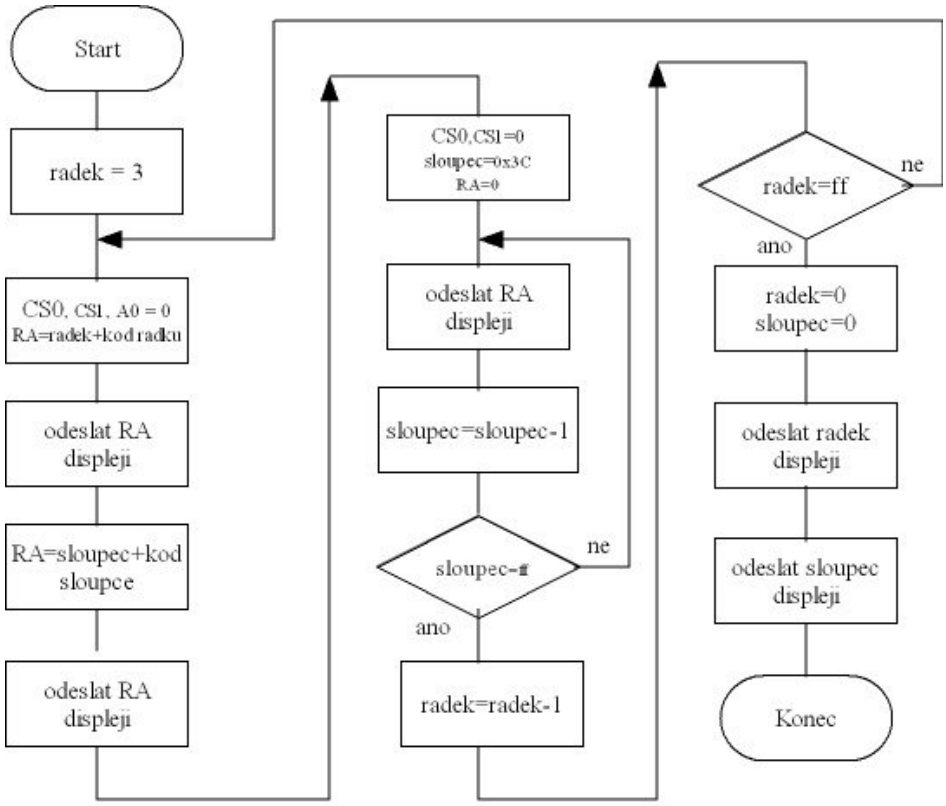
Obr. 27: Vývojový diagram algoritmu pro čtení znaku z ROM

Ve vývojovém diagramu je v úvodu omezeno vstupní číslo na rozsah 0 až 16 násobením maskou 0x0f. Následuje násobení 8x. Díky tomu, že lze maskou rozdělit vrcholovou adresu znaku od čísla aktuálního sloupce znaku, jde snadno testovat odeslání posledního sloupce znaku, aniž by byla potřeba další pomocná proměnná. Po získání čísla posledního odeslaného sloupce program otestuje, jestli už nejsou odeslány všechny sloupce znaku. Pokud jsou další sloupce na čtení, odešle se adresa sloupce paměti ROM a po malém zpoždění se přečtou aktuální data z paměti ROM, která odpovídají nastavené adrese. Po přečtení dat se adresa sloupce zvýší o 1. Přečtená data se uloží na displej. Jak je uvedeno výše je nejvhodnější využít již hotové funkce automatický posuvník, aby nebylo potřeba ještě kontrolovat, jestli není nutné přejít na druhou půlku displeje. Vývojový diagram není zcela přesný, ale s předchozím vysvětlením způsobu vracení z automatického posuvníku zpět do smyčky vkládání znaku by to nemělo vadit. Po přečtení všech sloupců z paměti ROM se musí vynulovat registr RD, aby nedošlo k chybnému vyhodnocení v automatickém posuvníku.

Funkce vynulování obsahu displeje

Pro rychlé smazání celého obsahu displeje slouží tato funkce. Funkce je bez parametrů, pouze se uloží nulovací kód do všech pozic displeje. Algoritmus je velmi podobný programu inicializace. Vývojový diagram algoritmu pro vymazání displeje je na Obr. 28.

V úvodu algoritmu se nastaví proměnná Radek na 3 a následně se vstoupí do smyčky, kde se nejprve odešle displeji adresa řádku a poté nulová adresa sloupce. Proběhne opakovaný zápis nulovacích dat, než dojde kurzor displeje na konec řádku. Následuje snížení proměnné Radek o 1 a celá smyčka se opakuje do vynulování této proměnné. V závěru funkce se vynulují proměnné Radek a Sloupec a nastaví se počátek displeje.



Obr. 28: Vývojový diagram algoritmu pro vymazání obsahu displeje

Celkový algoritmus zpracování příkazů s optimalizací

Z charakteru jednotlivých funkcí je zřejmé, že lze významně zjednodušit jednotlivé příkazy díky své podobnosti. Rozdíly mezi funkcemi pro vynulování řádku a sloupce, nový řádek a nastavení vlastního sloupce a řádku jsou minimální, pouze se v úvodu liší způsob nastavení proměnných Radek a Sloupec. Podobně i funkce automatický posuvník, při přechodu z jedné části displeje, vyžaduje přednastavení displeje podle aktuálního řádku a sloupce.

Funkce na vynulování obsahu displeje, díky své podobností s inicializací, lze realizovat skokem do části inicializace a po dokončení vymazání displeje program zůstane ve funkci na detekci taktovacího pulzu.

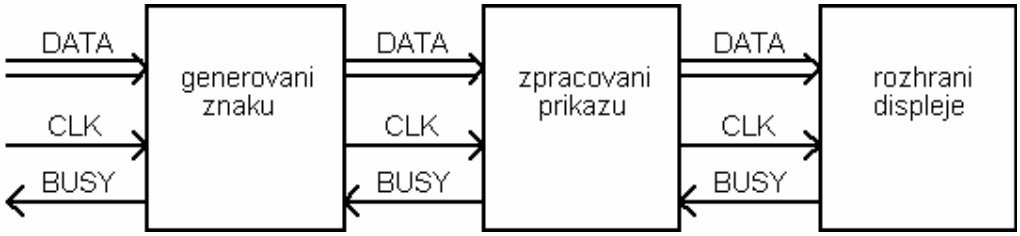
Navržený program

Blokový vývojový diagram celého programu je celkem jednoduchý. Skládá se z inicializace a nekonečné smyčky. Nekonečná smyčka obsahuje blokující funkci pro detekci náběžné hrany vstupního taktovacího pulzu a blok pro zpracování a vykonání příkazu. Popis funkce všech bloků je uveden v předchozí kapitole. Obr. 29 znázorňuje blokový diagram celého programu.

Obr. 29: Vývojový diagram celého programu

5. Blok řízení displeje realizovaný stavovým automatem

Blok řízení displeje realizovaný procesorem PicoBlaze má vyšší nároky na paměť a do malých obvodů, především obvodů CPLD, je problém jej implementovat. Obvykle by už nezbylo volné místo na laděnou aplikaci, proto byl realizován blok řízení displeje i stavovým automatem, který by měl být daleko jednodušší na implementaci. Aby výsledný stavový automat nebyl příliš složitý a mohly se využít jen ty funkce, které jsou pro danou aplikaci potřeba, byl blok řízení rozdělen do několika základních bloků, které na sebe navazují. Použitý LCD displej je proti taktování FPGA obvodu pomalý. Byl tedy vytvořen jednoduchý systém komunikace mezi bloky. Předpokládá se, že bloky budou postupně na sebe navazovat. Vznikne takový řetěz stavových automatů. Každý blok, kromě vstupního taktování, kterým se začnou zpracovávat vstupní data, musí mít i další vstup od následujícího bloku, kterým dává předchozí blok zprávu, že právě pracuje a nemůže přijímat další data. Tento signál byl pojmenován BUSY. Pak, pokud následující blok nastavil výstup BUSY a nemůžou se posílat další data, aktuální blok také nastaví výstup BUSY. Postupným předáváním signálů BUSY mezi bloky dojde informace o aktivitě bloků až na začátek řetězce. Potom jakákoliv část z řetězce řízení displeje může zastavit přicházející data na vstupu celého řetězce. Tato vlastnost je velká výhoda, protože snižuje nároky na výstup zdroje dat. Umožní zapsání potřebných dat, jak nejrychleji to jde. Protože zápis každého sloupce netrvá stejnou dobu, tak by se musela při uvažování periodického zápisu dat bez zpětné vazby rychlost taktování snížit na takovou velikost, jako by každý takt byl nejhorší případ. Pro názornost bylo vytvořeno principielní schéma řetězce, které je na Obr. 30.

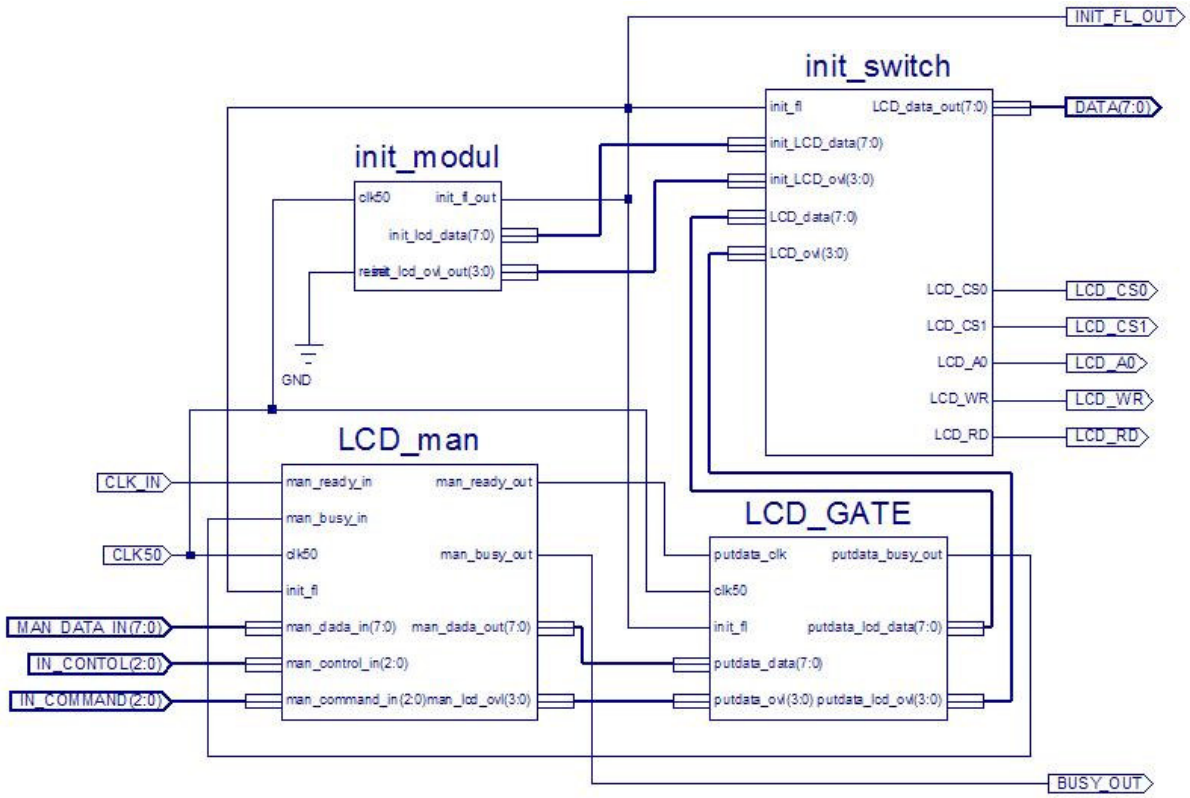


Obr. 30: Principielní schéma navrhovaného řetězce

Tento systém potvrzování má i malou nevýhodu. Protože předávání dat mezi bloky je zajištěno vzestupnou hranou, dochází při průchodu každým blokem ke zpoždění minimálně jeden takt. Taktovací pulz musí postupně dorazit až na konec řetězce. Až ten může odpovědět signálem BUSY, který se zase postupně předává zpět na začátek řetězce. Vzniká několika taktové zpoždění mezi odesláním platných dat a zpětného potvrzení signálu BUSY. Velikost zpoždění přímo závisí na počtu bloků v řetězci. Proto byla snaha návrh celého stavového automatu rozdělit na co nejméně bloků. Zdroj dat pak musí po odeslání dat počkat několik taktů, než se vrátí potvrzovací signál BUSY a až po jeho sestupné hraně může odesílat další data. S podobným zpožděním musí počítat i jednotlivé bloky řetězce. Zpoždění lze zkrátit tak, že každý blok po přijetí taktovacího pulzu nastaví svůj výstupní signál BUSY na 1, protože dříve nebo později bude nastaven vstupní signál BUSY do vysoké úrovně. Takové zrychlení může být nevýhoda. Při vložení kruhové vyrovnávací paměti mezi bloky, která odpoví signálem BUSY až po zaplnění její celé kapacity a předchozí bloky čekající na vzestupnou hranu signálu BUSY, dochází k poklesu rychlosti plnění takovéto paměti. Při každém zápisu do paměti se spustí nový timeout, který není zkrácen odesláním signálu BUSY.

5.1. Schéma základní struktury

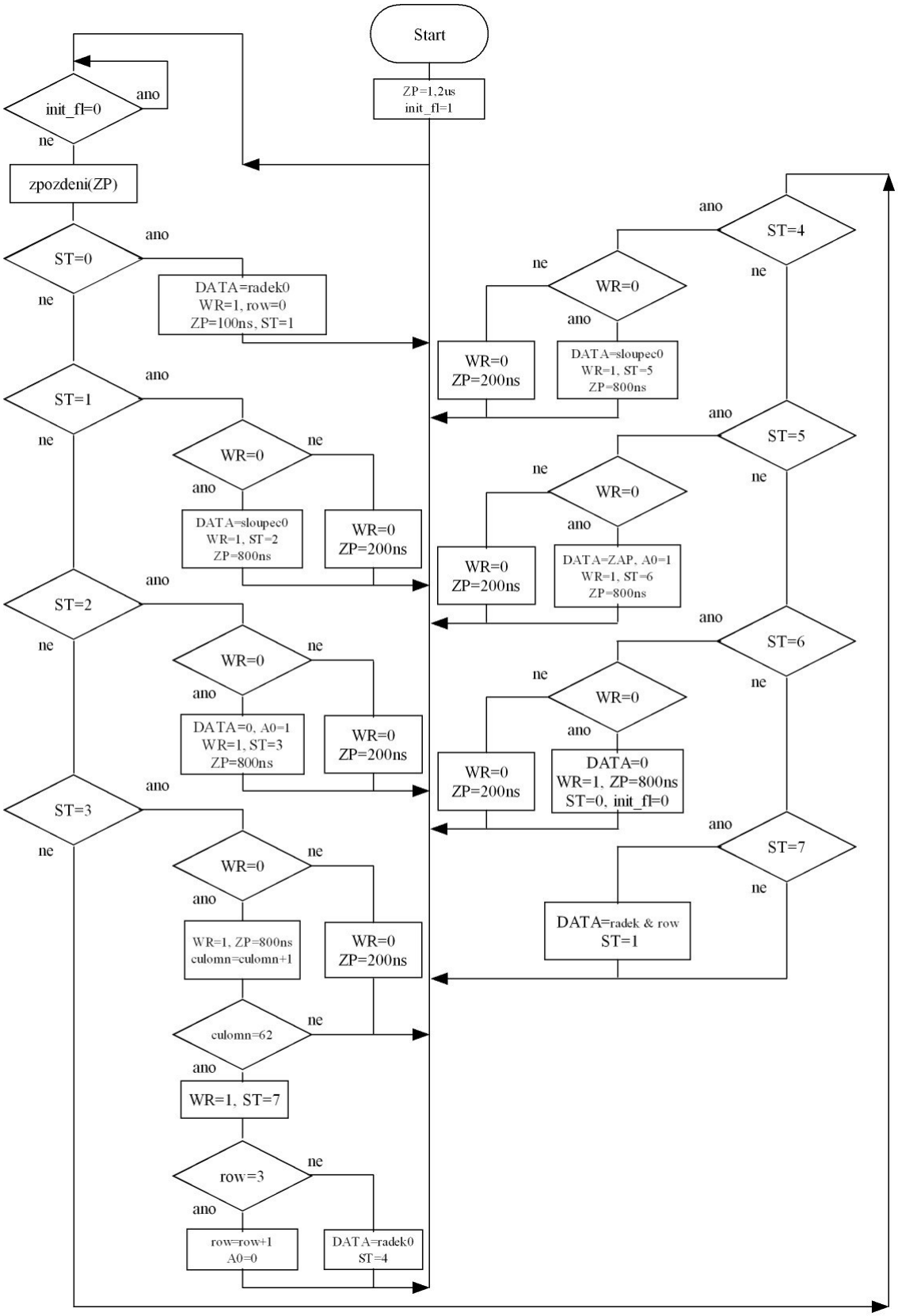
Základní struktura zajistí inicializaci a jednoduché ovládání displeje. Struktura se skládá z modulu inicializace, modulu pro vykonávání příkazů, rozhraním bloků s LCD displejem a přepínačem pro přepojení displeje po dokončení inicializace. Blok inicializace je navržen tak, že přímo komunikuje s displejem. Jakmile je displej zinicializován, přestane být modul inicializace aktivní a je nutné tento blok odpojit. Místo toho přepínač vybere signály od bloku LCD brány, která zůstane stále připojena. Přepínač reaguje na vstup init_fl. Protože navržené stavové automaty nepředpokládají čtení dat z displeje, byl signál RD trvale připojen na logickou úroveň 1. Podobně byl signál reset trvale uzemněn pro volbu typu komunikace s displejem jako u procesorů řady 80. Bylo by možné umístit přepínač před LCD bránu, ale to by se musel přepínat větší počet spojů. Potom by musel modul inicializace přijímat signály BUSY a generovat taktovací pulzy. Byl zvolen přepínač před displejem, aby bylo možné použít modul inicializace samotný bez nutnosti připojení dalších bloků. Navržené schéma zapojení bloků je na Obr. 31.



Obr. 31: Navržené schéma propojení bloků

Blok inicializace

Tento blok obsahuje pouze resetovací vstup pro opětovné zahájení inicializace a taktovací vstup. Pokud bude taktovací frekvence 50 MHz, budou dodrženy všechny časové zpoždění během komunikace s displejem dané katalogovým listem [8]. Díky zjednodušení ve výstupním přepínači, je pro komunikaci s displejem dostačující osmibitová datová sběrnice a čtyřbitová řídící sběrnice. Posledním výstupem bloku pro inicializaci je init_fl_out, kterým by měly být blokovány všechny ostatní funkce. Diagram stavového automatu je na Obr. 32.



Obr. 32: Diagram stavového automatu bloku inicializace

Celý algoritmus je realizován procesem. Zde je počáteční inicializace, která se provede po resetu celého programovatelného pole. Celý kód se cyklicky opakuje. Proto lze kód blokovat jen tak, že otestuje stav signálu init_fl. Pokud je nulový proces vykonává se pouze

jeho cyklické testování. Když tento signál má hodnotu 1, opakovaně se vstupuje do stavového automatu. Před vykonáním nastavení proměnných daných aktuálním stavem automatu se vykoná zpoždění dané proměnnou ZP. První zpoždění, které nastane po zapnutí napájecího napětí, je asi 1μs. Bylo odzkoušeno, že displej je schopen při takto malém zpoždění zpracovávat data. Pro větší zpoždění by se musel zvětšit počet bitů proměnné ZP a tím by se zvětšila velikost výsledného kódu.

V prvním stavu automatu se jen nakonfigurují porty, aby displej měl své vstupy v korektních stavech. Každá komunikace s displejem je rozdělena do dvou částí. V jedné části je taktovací signál WR v nulové úrovni a v druhé části má tento signál vysokou úroveň. Pro snížení počtu stavů automatu bylo této vlastnosti využito. Během vysoké úrovně signálu WR se může měnit stav sběrnic. Při prvním průchodu nějakým stavem automatu je signál WR ve vysoké úrovni a provede se pouze přiřazení nuly WR signálu. Po uplynutí zpoždění se změní stav automatu, přednastaví se nová data na sběrnicích a současně se signál WR nastaví do vysoké úrovně. Tím každému stavu automatu přísluší jedna dokončená komunikace.

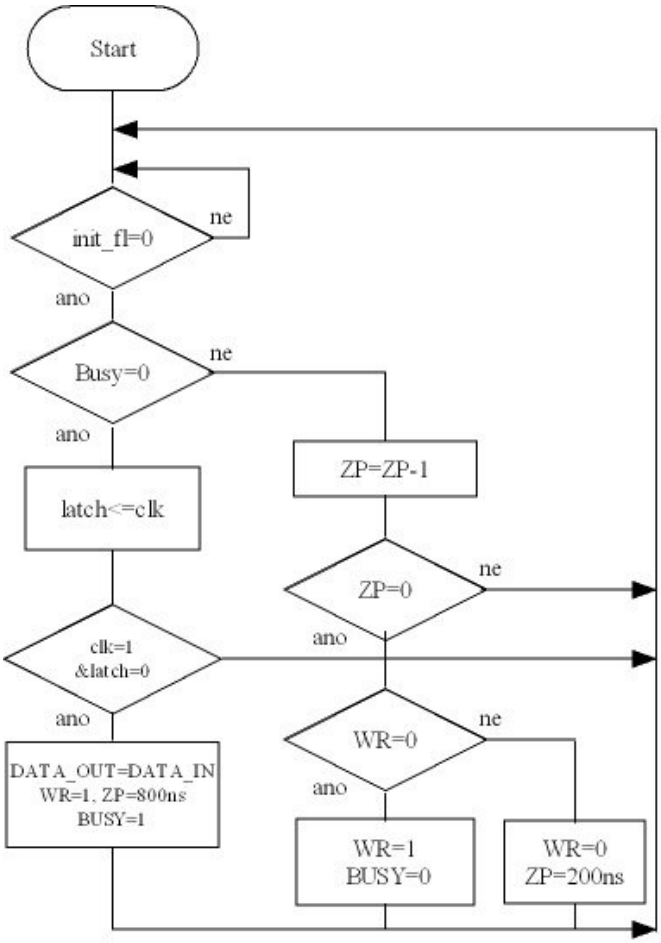
Ve stavovém automatu je řešeno vymazání celého obsahu displeje. Tuto činnost má na starost stav 3. V tomto stavu se testuje, jestli displej nedošel na konec řádku. Pokud ano, inkrementuje se číslo řádku. Když číslo řádku není 3, provede se přednastavení a nastaví se stav 7, který nastaví sběrnici pro nastavení dalšího řádku. Ze stavu 7 se přejde na stav 1, který opět uloží přednastavená data a nastaví adresu nulového sloupce. Po dosažení třetího řádku se opět kurzor displeje nastaví na počátek a příkazem ZAP se zapne zobrazování displeje. Před opuštěním stavu 6 se vynulují sběrnice a signál init_fl se vynuluje, aby se celý algoritmus zablokoval.

Vstupní resetovací signál je testován jen v případě nulového signálu init_fl. Je tu opět vytvořen latchový signál pro zapamatování předchozího stavu signálu reset a následnou detekci vzestupné hrany signálu reset.

Blok zápisu dat displeji

Tento blok představuje rozhraní mezi displejem a ostatními bloky ovladače. Algoritmus dodržuje časovou posloupnost ukládaných dat dle doporučení katalogového listu [8]. Blok obsahuje osmibitový vstup dat dále 3 bity pro konfiguraci displeje a taktovací pulz, který potvrzuje zápis vstupních dat na displej. Pokud tento blok pracuje, nastaví svůj výstup BUSY na vysokou úroveň. Předpokládá se připojení taktovacího signálu s kmitočtem 50 MHz na vstup clk50. Vstup init_fl slouží pro blokování této funkce v době inicializace. Vývojový diagram funkce pro zápis dat displeji je na Obr. 33.

Celý algoritmus je umístěn v procesu. Z toho plyne, že algoritmus musí být navržen jako nekonečná smyčka. Blokování funkce vstupem init_fl je realizováno tak, že pokud je nenulový, opakovaně se testuje, jestli se nezměnila hodnota na 0. Po odblokování funkce se testuje, jestli byl nastaven signál BUSY. Pokud ne, čeká se na vzestupnou hranu. Proto byl vytvořen signál latch, který uchovává předchozí stav signálu CLK. Pokud platí, že latch=0 a současně CLK = 1, znamená to, že nastala vzestupná hrana a je nutné uložit vstupní data displeji. V tomto stavu je nutné nastavit signál BUSY do vysoké úrovně. Protože byla snaha o co nejjednodušší výsledný algoritmus, bylo trochu pozměněno časování. Aby nevadilo, že signál WR není před začátkem zápisu ve vysoké úrovni a při zahájení zapisování nedošlo k nekorektnímu zápisu, byla doba čekání mezi předchozím a následujícím zápisem umístěna před vlastní zápis. Proto jsou nejprve nastaveny datové a řídící sběrnice a nastavena proměnná ZP na 800 ns. Zápis je proveden obdobně jako v bloku pro inicializaci. Po dokončení zápisu se musí vynulovat výstupní signál BUSY.

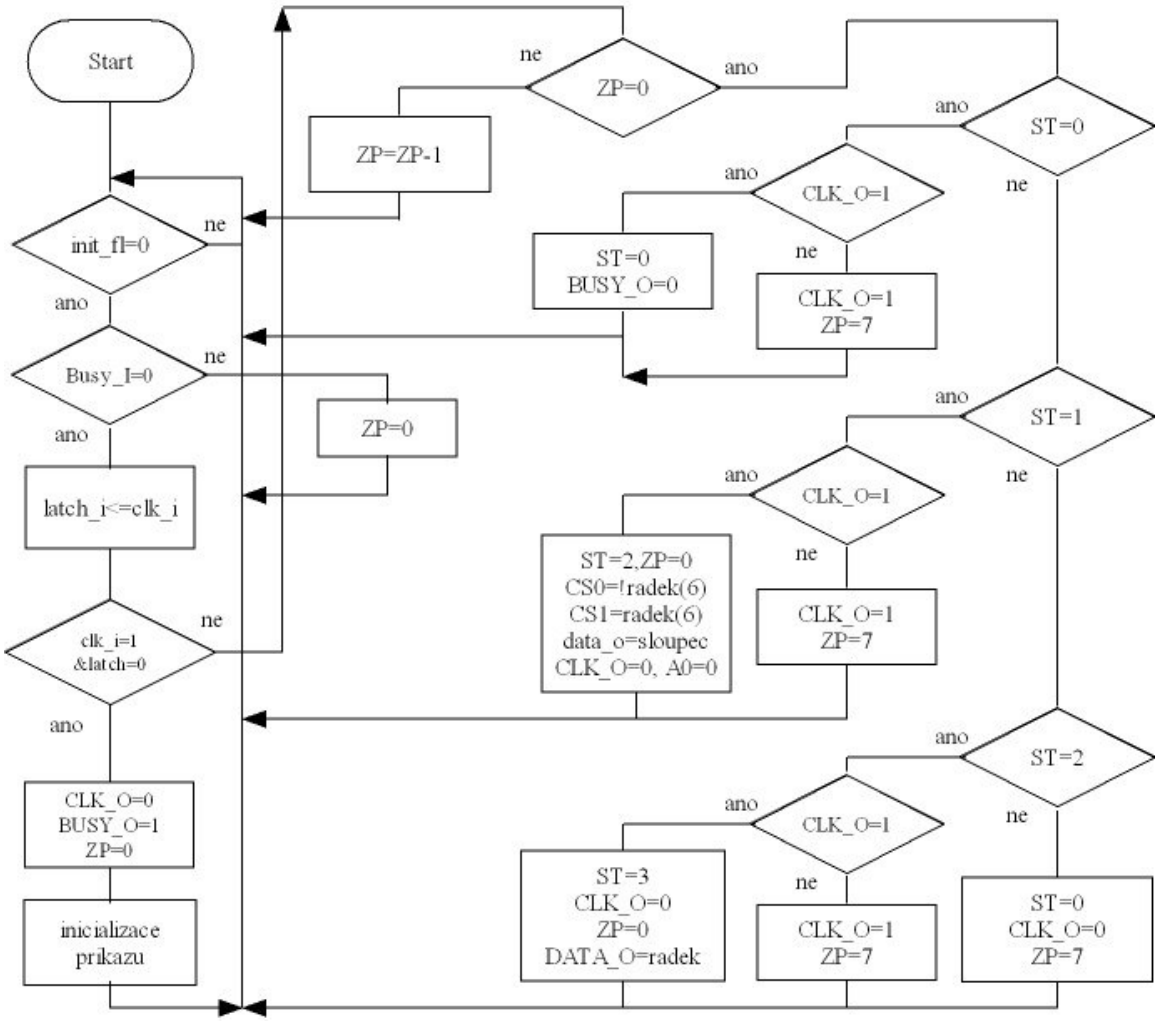


Obr. 33: Vývojový diagram bloku pro zápis dat displeji

Blok zpracování příkazů

Hlavní funkce bloku pro zpracování příkazů, jak je z názvu patrné, zpracuje vstupní data podle nastavené příkazu. V tomto bloku jsou zahrnuty podobné funkce, jaké jsou v návrhu s procesorem PicoBlaze. Mezi nejzajímavější patří automatický posuvník. Dále je tu funkce pro nastavení nulového sloupce a řádku, nastavení nového řádku a nastavení libovolného řádku a sloupce. Funkce pro vymazání displeje nebyla implementována, protože lze displej vymazat resetováním bloku inicializace. Dále pro zjednodušení celého stavového automatu byla funkce pro vkládání symbolů z paměti ROM realizována jako jeden samostatný blok.

Řízení bloku pro zpracování příkazů je pomocí vstupů CLK a BUSY. Tento blok už obsahuje vstupní i výstupní CLK i BUSY signál. Jak bylo popsáno výše, je tu dodržen systém, že vstupní CLK dává informaci o tom, že na vstupu jsou platná data a výstupní CLK předává data následujícímu bloku. Vstupní signál BUSY blokuje činnost algoritmu pro zpracování signálu a výstupní signál BUSY je pro předchozí blok, aby neposílal další data. Mezi další vstupy patří osmibitová datová sběrnice, po které se předávají data, tříbitová sběrnice pro volbu příkazu a tříbitová sběrnice pro předání konfigurace displeje při přímém posílání dat. Výstup předává pouze osmibitová data a tříbitovou konfiguraci displeje. Na Obr. 34 je vývojový diagram bloku pro zpracování příkazů.



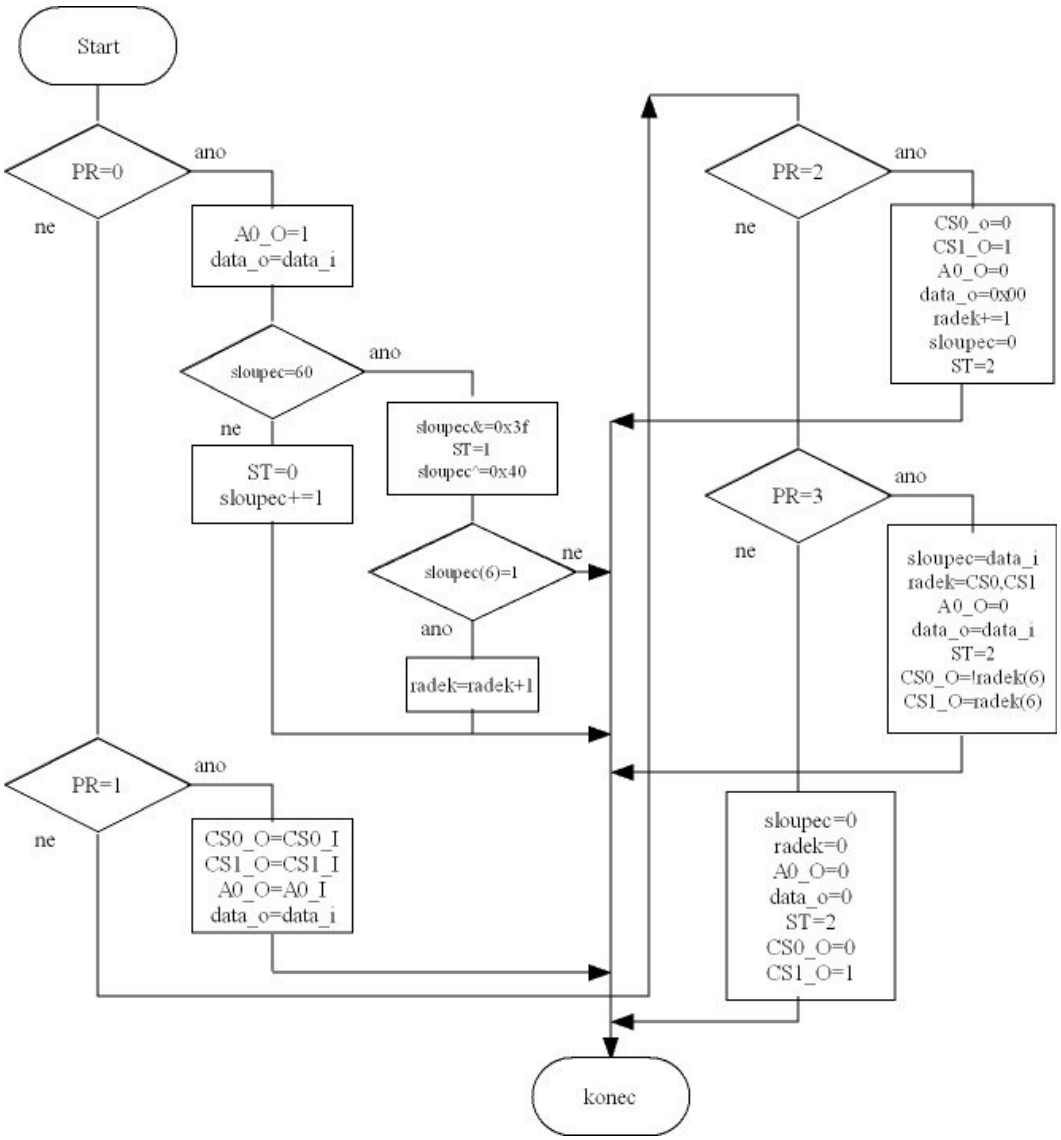
Obr. 34: Vývojový diagram bloku pro zápis dat displeji

Blok pro zápis dat bylo nejvýhodnější realizovat procesem. Algoritmus je v úvodu doplněn o blokování vstupem init_fl. Algoritmus dělí své chování podle toho, v jakém stavu je vstupní signálu BUSY_IN. Pokud je nastaven, proces nesmí předávat displeji data. Protože zde vzniká prodleva mezi náběžnou hranou výstupního taktovacího pulzu a potvrzovacím pulzem BUSY_IN, je před každou náběžnou hranou pulzu inicializováno zpoždění zapsáním nenulové hodnoty do proměnné ZP. Během nulové úrovně signálu BUSY_IN se místo přechodu k dalšímu kroku stavového automatu pouze snižuje proměnná ZP s každým cyklem o 1. Pokud by se hodnota proměnné snížila na 0 automat by odeslal další data. Proto se musí tato inicializační hodnota volit podle množství následujících bloků. Po nastavení vstupu BUSY_IN se proměnná ZP vynuluje. Tento princip čekání na potvrzovací úroveň signálu a blokování stavového automatu lze využít i v dalších částech celého řetězce, jen je potřeba pro každý blok zvážit, jak velké zpoždění je potřeba. Zbytečně velkým zpožděním se zvyšuje velikost celkového kódu. Dále, v případě připojení k bloku bez generování signálu BUSY_IN, by docházelo ke zbytečnému prodlužování komunikace, protože by stavový automat vydával data pomaleji.

K dalším velkým rozvětvením dochází detekcí vzestupné hrany vstupního taktovacího signálu. Po detekci vzestupné hrany se provede přednastavení podle aktuální předvolby dané vstupem man_command_in. Protože všechny příkazy jsou do jisté míry podobné, jejich následné vykonávání může využít totožné funkce. Vykonávání příkazů lze provést stavovým automatem. Pak volbou stavu automatu se nastavená data zpracují. Stavový automat bude

generovat taktovací pulz po nastavení datové a kontrolní sběrnice. Taktovací pulz představuje půlku doby ve vysoké úrovni a druhou půlku v nízké úrovni. Toho je využito při zjednodušování stavového automatu, protože pro každou úroveň stavového automatu je vygenerován celý taktovací pulz. V každé úrovni stavového automatu je proto test hodnoty taktovacího signálu.

Stav automatu 0 zajistí pouze správné odeslání přednastavených dat následujícímu bloku s vyčkáním do nastavení signálu BUSY_IN na nízkou úroveň. Stav automatu 2 spolu se stavem 3 slouží pro nastavení pozice kurzoru na displeji. Předpokládá se, že řídící i datové sběrnice jsou nastaveny. Prvním vkročením do stavu 2 se vygeneruje vzestupná hrana taktovacího pulzu a současně se přednastaví proměnná ZP. Po změně signálu BUSY_IN na nulu se přednastaví požadovaný řádek na datové sběrnici a přejde se na další stav automatu. Ten přejde na stav 0 automatu. Stav 0 pak dokončí odeslání řádku displeji. Aby vývojový diagram funkce na zpracování příkazu byl přehledný, byla část vývojového diagramu, ve které se nastavují sběrnice podle aktuálního příkazu, oddělena. Rozpoznání příkazu s nastavením je na Obr. 35.

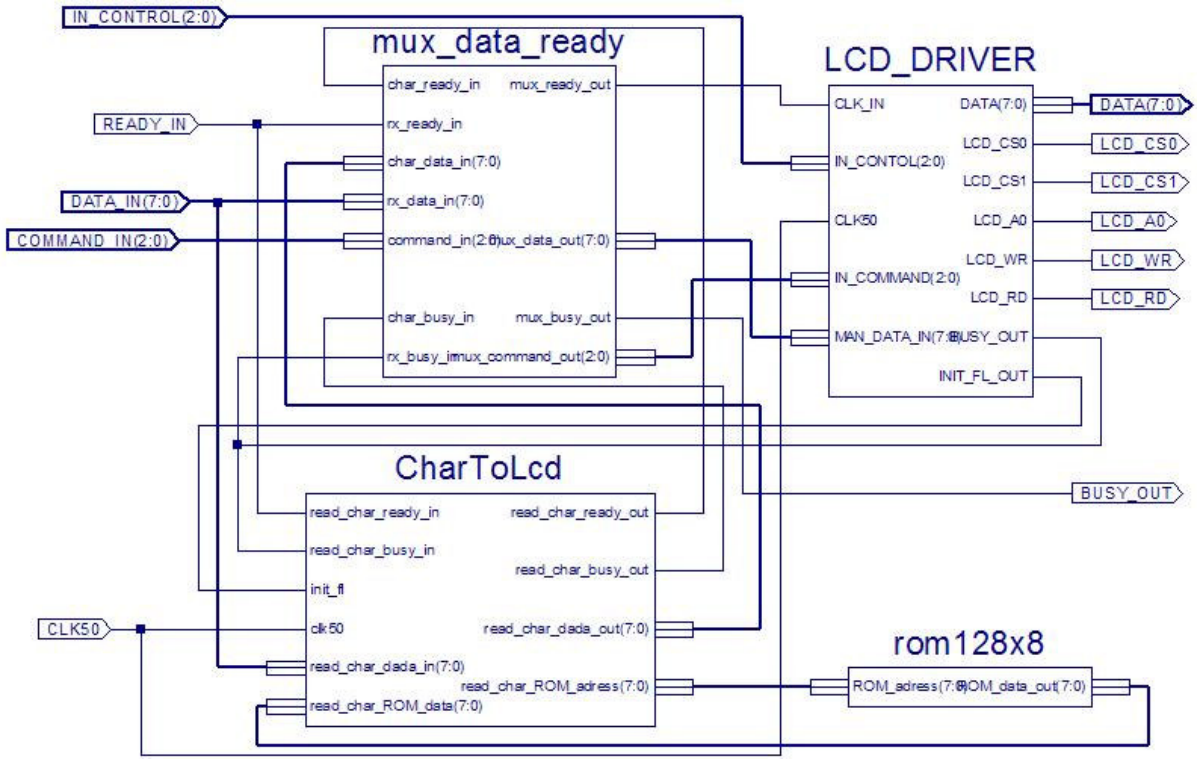


Obr. 35: Vývojový diagram bloku pro rozpoznání příkazu

Funkce na rozpoznání příkazu přiřazuje kódu 0 funkci automatického posuvníku. Ten pouze zapisuje data na displej. Při běžném ukládání se přednastaví úroveň stavového automatu nulu a data se pouze odešlou na výstup. Když zapsaná data jsou na posledním sloupci v řádku, hodnota sloupce se vynuluje a číslo řádku se zvýší o 1. Současně se předvolí stavový automat na úroveň 1, který data zapíše a následně posune kurzor displeje do počátku nového řádku. Příkaz s kódem 1 přesune vstupní data na výstup a předvolbu kontrolní sběrnice použije ze vstupu. Příkaz s kódem 2 posune kurzor na počátek následujícího řádku. Obdobnou funkci má klávesa ENTER. Příkaz s kódem 3 dovolí nastavit kurzor kamkoliv na displej. Na vstupu DATA_IN číslo o novém sloupci a vstupy CS0 a CS1 představují kód řádku. Ostatní kódy nastaví kurzor na počátek displeje, do levého horního rohu.

5.2. Čtení znaku z paměti ROM

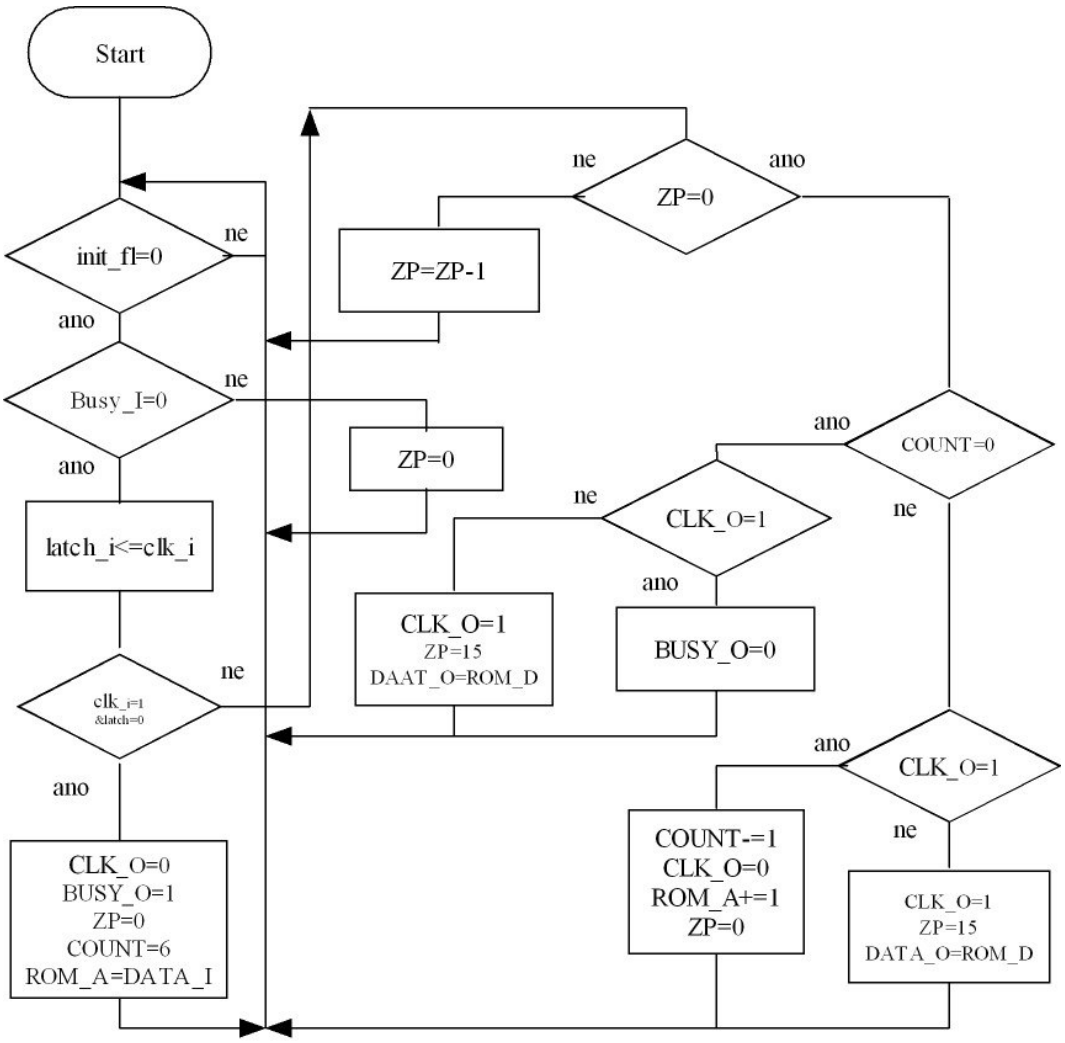
Předchozí blok neobsahoval funkci na převod vstupních dat na symbol. Tato funkce nemusí být vždy potřeba a příliš by zvyšovala složitost předchozího bloku. Na Obr. 36 představuje blok s názvem LCD_DRIVER předcházející popisovaný blok. Vlastní čtení znaků z paměti ROM zajišťuje blok s názvem CharToLcd. Aby se mohly využít všechny funkce bloku LCD_DRIVER, byl do schématu přidán jednoduchý přepínač, kterým se volí zdroj kontrolních a datových signálů. Přepínač, skoro pro všechny kódy příkazové sběrnice, propojuje vstupní data přímo k LCD ovladači. Pouze pro hodnotu 6 začne být zdrojem kontrolní i datové sběrnice blok CharToLcd. K tomuto bloku je připojena paměť ROM, kterou je dán font zobrazovaných symbolů. Předpokládá se stejná organizace paměti jaká byla použita v procesoru PicoBlaze.



Obr. 36: Schématické vyjádření bloku se zápisem znaků z paměti ROM

Blok čtení dat ze znakové paměti ROM

U tohoto bloku byl dodržen systém taktování CLK a systém zpětné vazby signálem BUSY. Blok vyžaduje taktovací frekvenci 50MHz, vstup CLK a vstup DATA_IN, kterým se zadávají kódy znaků. Dále je přiveden datový vstup z paměti ROM a také generuje sedmi bitovou adresu pro paměť znaků. Výstupem DATA_OUT spolu s CLK_OUT (READY_OUT) se předávají ovladači displeje jednotlivé sloupce znaků. Samozřejmě signálem BUSY_OUT informuje bloky o své aktivitě. Vývojový diagram algoritmu pro čtení z paměti ROM je na Obr. 37.



Obr. 37: Vývojový diagram algoritmu pro čtení znaků z paměti ROM

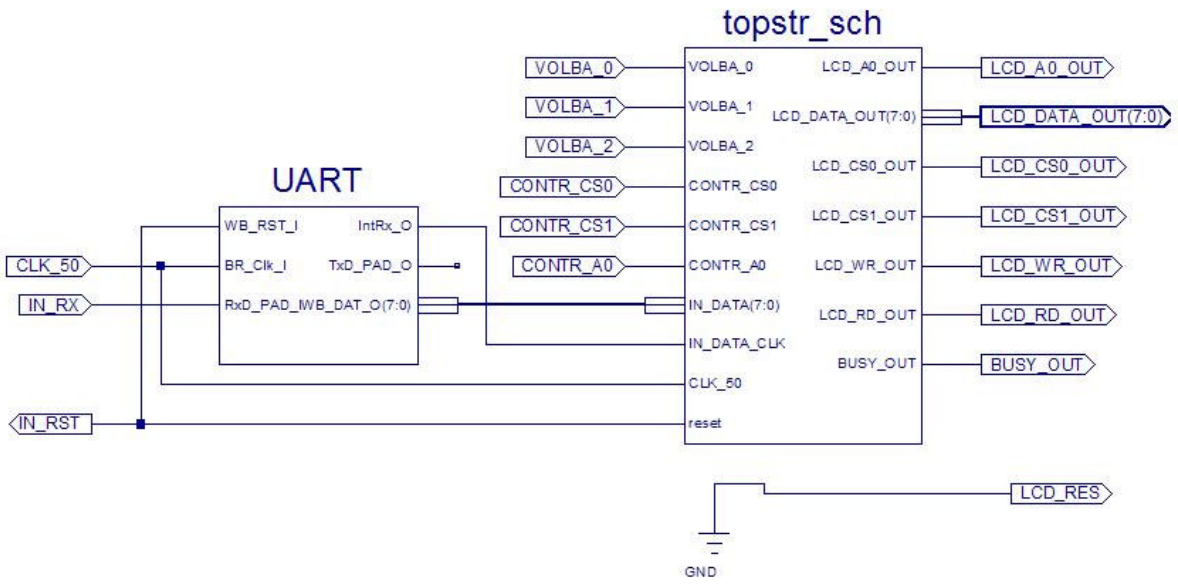
Navržený algoritmus se podobá algoritmu pro zpracování příkazu, protože jeho struktura vychází z požadavků chování na taktovací pulzy CLK_IN a potvrzovací signál BUSY_IN. Ve struktuře je pouze změněna inicializace a pak následně způsob zpracování a odeslání dalších dat. V inicializaci se nastaví počet vyčítaných byte v jednom symbolu a nastaví se vrcholová adresa paměti ROM. Při první vzestupné hraně signálu CLK_O se přečtou data z paměti ROM a odešlou se na výstup. Současně se nastaví velikost zpoždění ZP = 15. Hodnota ZP má větší hodnotu proti předchozímu bloku, protože předchozí blok vytvoří více taktové zpoždění. Po úspěšném odeslání dat až k displeji se sníží hodnota proměnné COUNT o 1 a zvýší se adresa paměti ROM o 1. Následující takt opět generuje náběžnou hranu na signálu CLK_O a ten opět zapíše data displeji. Takto algoritmus pokračuje

až do snížení proměnné COUNT na nulu. Zde se odešlou poslední přečtená data z paměti ROM a v závěru se nastaví signál BUSY_OUT na nulu. Tím se dokončí komunikace a blok je schopen přijímat a zpracovávat další data.

5.3. Příklad použití

Navržený ovladač LCD displeje pouze usnadňuje a zjednodušuje komunikaci. Aby použití ovladače bylo co nejjednodušší a nejuniverzálnější, byl vytvořen pouze ovladač s jednoduchým ovládáním. Bylo by možné ovladač rozšířit o algoritmus periodického umisťování dat do určitého místa na displeji pro snadnější sledování proměnných v laděných procesech, ale to by příliš svazovalo celý ovladač na konkrétní aplikaci. Vytvoření takového algoritmu je velmi snadné při použití struktury pro generování signálu CLK_O a BUSY_O, které jsou použity v bloku pro zpracování příkazů a bloku pro vyčtení znaků z paměti ROM. Algoritmus neomezuje počet sériově připojených bloků, pouze s každým blokem vzniká zpoždění několik taktů generátoru 50 MHz, se kterým se musí počítat.

Pro snadné ověření implementovaných funkcí byl k celé navržené struktuře připojen blok UART, který převádí sériová data na paralelní osmibitová slova. Další informace k použitému bloku UART je v [15]. Testování bylo provedeno na přípravku Spartan 3, který je vybaven tlačítky a led diodami. Blok UART přijímá sériovou komunikaci a po odeslání dat blok umístí data na svůj výstup a současně vygeneruje vzestupnou hranu výstupního taktovacího signálu. Data jsou předána ovladači LCD displeje a ten je zpracuje podle předvolby. Předvolba ovladače je vyvedena na přepínačích. Schéma testovací struktury je na Obr. 38: Schéma testovací struktury s modulem UART.



Obr. 38: Schéma testovací struktury s modulem UART

Stejným způsobem byla otestována i realizace s procesorem PicoBlaze, protože má stejné řídící i datové sběrnice. Oba způsoby řešení jsou ekvivalentní. Pouze se liší rychlostí zpracování přijatých dat. Realizace s procesorem PicoBlaze je pomalejší, protože se vykonávají instrukce a to je daleko pomalejší než stavový automat.

6. Závěr

LCD displeje se mohou rozdělit na alfanumerické a grafické. Alfanumerické displeje mají ve vnitřní paměti uloženou znakovou sadu, proto se text píše jednodušeji. Je u nich komplikovanější zobrazovat symboly, které nejsou součástí znakové sady. Popisované grafické displeje nemají paměť ve které je znaková sada. Proto se musí každý symbol definovat v ovládacím programu. Jejich výhoda zase spočívá v tom, že není problém zobrazit libovolný znak.

Přídavná deska byla navržena pro displej s označením GM12321. Tento displej má rozhodovací úrovně jako TTL. Obvody CPLD nebo FPGA, ke kterým se přídavná deska připojí, mají výstupní úrovně technologie LVCMOS33. Do tolerance vstupního napětí pro platnou úroveň H displeje spadají i výstupní velikosti napětí technologie LVCMOS33. Proto navrhované schéma nepotřebuje aktivní převodník napětí a stačí pasivní realizovaný rezistory. Dále přídavná deska musí obsahovat oscilátor s frekvencí 2 kHz a zdroj záporného napětí, aby LCD displej mohl zobrazovat údaje.

Nejprve byla přídavná deska testována na vývojovém kitu XC2-XL, ale obvod XC2C256 byl příliš malý na implementaci ovládacího programu s procesorem PicoBlaze. Proto se přídavná deska testovala na vývojovém kitu Spartan 3.

Do programu v procesoru PicoBlaze bylo implementováno několik funkcí pro základní konfiguraci displeje a pro jeho snazší ovládání. Dále byla implementována funkce automatického posuvníku, která zajistí přechod mezi částmi displeje bez zásahu uživatele. Další implementované funkce jsou pro snadné nastavení kurzoru na displeji a vymazání obsahu displeje. Velkým přínosem je i funkce pro převod vstupních dat na znaky. Znaky jsou uložené v externí paměti ROM, Snadnou výměnou této paměti lze změnit znakový font. V návrhu byl vytvořen malý font pro vyjádření hexadecimálních čísel. Procesor po přijetí příkazu generuje na svém výstupu signál BUSY, kterým informuje předchozí blok o své aktivitě a proto předchozí blok nemá odesílat další data. Program je rozdělen do tří základních částí. Inicializace vymaže interní RAM displeje. Detektor taktovacího pulzu odblokuje celý program při vzestupné hraně taktovacího signálu a dovolí vykonání nastaveného příkazu blokem pro zpracování příkazu.

Kromě realizace ovladače LCD displeje procesorem PicoBlaze byl vytvořen ovladač pomocí stavového automatu v kódu vhdl. Ovládání displeje stavovým automatem je skoro totožné jako v návrhu s procesorem PicoBlaze. Liší se pouze ve funkci resetování displeje. U procesoru je reset proveden příkazem, ale u stavového automatu se reset provede vzestupnou hranou resetovacího vstupu. Byl vytvořen blok inicializace displeje, který po zapnutí napájení vymaže obsah interní paměti RAM displeje a připraví displej pro další používání. Dále byl vytvořen blok pro komunikaci s displejem. Ten vytváří rozhraní mezi displejem a ostatními bloky návrhu. Zajistí také správné odeslání dat displeji. Byl vytvořen systém předávání a potvrzování dat, který zjednodušuje komunikaci s displejem a zabrání příliš rychlému přísunu dat dříve, než je displej schopen je přijmout. Blok pro zpracování příkazu provádí základní operace s kurzorem displeje. Také je tu vytvořena funkce automatického posuvníku jako u procesoru PicoBlaze. Pro zápis symbolů z paměti ROM byl vytvořen další blok CharToLcd. Ten v aktivním stavu převede vstupní data na posloupnost dat, kterou odešle displeji. Organizace paměti ROM je stejná jako u procesoru PicoBlaze.

Pro ověření implementovaných funkcí byl na vstup navržených modulů připojen UART modul, který převádí sériová data z počítače na testovaný blok.

7. Seznam literatury

[1] KOLOUCH, J. *Programovatelné logické obvody a návrh aplikací v jazyku VHDL – Počítačové cvičení*. Brno: Vysoké učení technické v Brně, 2005. 85 stran. ISBN 80-214-1997-6.

[2] KOLOUCH, J. *Programovatelné logické obvody*. Brno: Vysoké učení technické v Brně, 2005. 97 stran.

[3] MATOUŠEK, D. *Práce s mikrokontrolery ATMEL AVR*. BEN – Technická literature Praha, 2003.

[4] XILINX. *SPARTAN-3 FPGA – dokumentace*. Dostupný na WWW: <http://www.xilinx.com/products/silicon_solutions/fpgas/spartan_series/spartan3_fpgas/>

[5] DIGILETIC. *XC2-XL – dokumentace*, Dostupný na WWW: <http://www.digilentinc.com/Products/Detail.cfm?Prod=XC2XL&Nav1=Products&Nav2=Programmable>

[6] DATA IMAGE CORPORATION. *Katalogový list CM1621*. Dostupný na WWW: <http://www.koala.cz/download/displeje/alphanumericke/cm1621s1ly-j3.pdf>

[7] HITACHI. *katalogový list řadiče HD44780*. Dostupný na WWW: <http://www.sparkfun.com/datasheets/LCD/HD44780.pdf>

[8] DATA IMAGE CORPORATION. *Katalogový list GM12321*. Dostupný na WWW: <http://www.koala.cz/download/displeje/graficke/gm12321as1ly-j1.pdf>

[9] S-MOS SYSTEMS. *Katalogový list řadiče SED1520*. Dostupný na WWW: <http://www.lcd-module.de/eng/pdf/zubehoer/sed1520.pdf>

[10] DATA IMAGE CORPORATION. Katalogový list GM12641. Dostupný na WWW: <http://www.koala.cz/download/displeje/graficke/gm12641sly-j3.pdf>

[11] SAMSUNG. K*atalogový list řadiče KS0107*. Dostupný na WWW: <http://www.ocularlcd.com/pdfs/driver/ks0107.pdf>

[12] XILINX. *Katalogový list vývojového kitu Spartan 3*. Dostupný na WWW: <http://www.xilinx.com/support/documentation/user_guides/ug332.pdf>

[13] KOLOUCH, J., BIOLOVÁ, V., JAKUBOVÁ, I. *Impulzová a číslicová technika-Laboratorní cvičení* Brno: Vysoké učení technické v Brně, 2005. 97 stran.

[14] KOLOUCH, J. *Typické postupy při práci s obvody PLD a FPGA*. Dostupný na WWW: <http://www.urel.feec.vutbr.cz/~kolouch/pld/data/JakNaToPldFpga.pdf>

[15] CARTON , C *MiniUART IP Core*. Dostupný na WWW: <http://www.opencores.org/projects.cgi/web/miniuart2/overview>